Sumari d'annexos

B.2.6. Exercici 5

Exercici 5. ***Fer que el robot camini indefinidament fins a trobar-se amb un obstacle que serà detectat mitjançant el sensor de ultrasons, moment en que s'aturarà. Posteriorment haurà de provar de desplaçar-se lateralment per comprovar si és un obstacle puntual al camí o no. En cas afirmatiu haurà de seguir avançant. En cas contrari, haurà de donar mitja volta i seguir caminant en sentit contrari.***

```
'Declaració de constants
CONST DIST_MIN = 25
CONST MAX_PASSOS = 4
CONST GIRS_ESQUERRA = 7
CONST MAX_LAT = 5
```

Es declaren les constants i els seus valors consegüents que es vol que tinguin.

```
'Declaració de variables
DIM ultrasons AS INTEGER
DIM obstacle AS BYTE
DIM passos AS BYTE
DIM girs AS BYTE
DIM laterals AS BYTE
```

Es declaren les variables de tipus byte (8 bits - [0-255]) i de tipus integer (16 bits – [0-65535]) per usar-les posteriorment en el programa.

```
'Configure PTP for using with groups of servos
PTP SETON
'Configure PTP for using with all the servos in the robot
PTP ALLON
```

Activa el funcionament punt a punt de tots els servomotors.

```
'== motor direction setting ====================
DIR G6A,1,0,0,1,0,0
DIR G6B,1,1,1,1,1,1
DIR G6C,0,0,0,0,0,0
DIR G6D,0,1,1,0,1,0
```

Es defineix el sentit de gir dels diferents servomotors. Si hi ha un 1 el servomotor girarà en sentit horari mentre que un 0 farà que el servo giri en sentit antihorari.

La definició de moviments i tractaments pels servos es fa sempre per 24 motors (dels quals realment només estan 16).

Grup 6A: Cama esquerra (5 servomotors)

Grup 6B: Braç esquerre(3 servomotors)

Grup 6C: Braç dret (3 servomotors)

Grup 6D: Cama dreta (5 servomotors).

```
'== motor start position read ====================
GETMOTORSET G6A,1,1,1,1,1,0
GETMOTORSET G6B,1,1,1,0,0,0
GETMOTORSET G6C,1,1,1,0,0,0
GETMOTORSET G6D,1,1,1,1,1,0
```

Definició de la posició inicial al iniciar el programa. En cas de ser un 1 els motors es quedaran a la posició que estaven, en cas de tenir un 0 es mouran a una posició de seguretat indicada per el controladors.

```
SPEED 5
```

S'assigna a la velocitat el valor 5. Es recomana no utilitza velocitats superiors a 10 per les diferents pràctiques a realitzar

```
'== motor power on ============================
MOTOR G24
GOSUB standard_pose

'================================================
'================================================

'Main procedure
MAIN:


	' Inicialització
	passos = 1
	obstacle = 0
	girs = 1
	laterals = 1

	DELAY 2000
```

S'inicialitzen les variables que es faran servir durant el programa i es fa una pausa de 2 segons abans de començar.

```
	'Captar el valor del sensor de ultrasons
	GOSUB capturar_valor_ultrasons


	' Primera comprovació de distancia i primer pas
	IF ultrasons < DIST_MIN THEN
		obstacle = 1
	ELSEIF ultrasons >= DIST_MIN THEN
		GOSUB primer_pas
	ENDIF
```

```
'Comprovació reiterativa de distancia i caminar
IF obstacle = 0 THEN
      FOR passos = 1 TO MAX_PASSOS
            IF obstacle = 0 THEN
                  GOSUB capturar_valor_ultrasons
                  IF ultrasons < DIST_MIN THEN
                        obstacle = 1
                  ELSEIF ultrasons >= DIST_MIN THEN
                        GOSUB caminar
                  ENDIF
            ENDIF
      NEXT passos
      GOSUB retorn_inici
ENDIF
```

Es capturar el valor del sensor de ultrasons. En cas que es trobi un obstacle frontal a una distància inferior a la mínima (calculada prèviament per tal que el robot pugui fer un pas complet), s'activa la variable obstacle. En cas contrari es farà el primer pas (primera part del moviment de caminar).

Posteriorment si no s'ha trobat obstacle es realitzaran diversos passos consecutius. Abans de fer cada pas, es capturarà el valor del sensor per veure si existeix obstacle. Finalment tant si es para el robot per un obstacle com si acaba de fer tots els passos tornarà a la posició estàndard. La utilització de dos sentències if i una for es degut a que el llenguatge RoboBasic no conté cap sentència while.

```
'Desplaçament lateral per provar d'evitar un obstacle puntual
DELAY 2000
IF obstacle = 1 THEN
      FOR laterals = 1 TO MAX_LAT
            GOSUB lateral_dreta
      NEXT laterals
      GOSUB capturar_valor_ultrasons
      IF ultrasons >= DIST_MIN THEN
            obstacle = 0
      ENDIF
ENDIF
```

En cas de tenir un obstacle lateral es realitzaran uns desplaçaments laterals cap a la dreta. Tot seguit es tornarà a comprovar si segueix existint obstacle frontal i en cas negatiu es desactivarà la variable obstacle.

```
'En cas de seguir existint obstacle donem mitja volta
IF obstacle = 1 THEN
      FOR girs = 1 TO GIRS_ESQUERRA
            GOSUB girar_esquerra
            GOSUB standard_pose
            DELAY 1000
      NEXT girs
ENDIF
```

En últim cas, si després de fer els desplaçaments laterals segueix havent un obstacle, es donarà mitja volta mitjançant una sèrie de girs cap a l'esquerra.

```
        'Tornem a reiniciar el programa sencer (ja que sempre seguirà
        caminant, simplement podrà canviar la direcció segons els
        obstacles)
        GOSUB standard_pose
        GOTO MAIN
```

Finalment, com el programa es reiniciarà de forma que el robot caminarà indefinidament. Malgrat això cal dir que cada MAX_PASSOS el robot tornarà a la posició estàndard i seguirà caminant després de 2 segons. D'aquesta forma es té un petit control de l'execució del programa.

```
        '================================================
        '================================================
standard_pose:
        SPEED 5
        MOVE G6A,100, 76, 145, 93, 100, 100
        MOVE G6D,100, 76, 145, 93, 100, 100
        MOVE G6B,100, 30, 80, 100, 100, 100
        MOVE G6C,100, 30, 80, 100, 100, 100
        WAIT
        RETURN
```

Es defineixen els valors de la posició estàndard (posició inicial i de seguretat).

```
    '================================================
    '================================================
capturar_valor_ultrasons:

        ultrasons = AD(2)
        ultrasons = ultrasons * 13
        ultrasons = ultrasons / 10

        WAIT
        RETURN
```

Es capta el valor del sensor de ultrasons i es transforma a cm fent servir les comandes explicades.

```
        '================================================
        '================================================
primer_pas:

        'Inclinar-se a la dreta
         SPEED 5
        MOVE24 85, 71, 152, 91, 112, 60, 100, 40,  80,    ,    ,    ,
        100,  40,  80,    ,    ,    , 112,  76, 145,  93,  92,  60,

        'Aixecar peu esquerre
        SPEED 10
        MOVE24  90, 107, 105, 105, 114,  60,  90,  40,  80,    ,    ,
        , 100,  40,  80,    ,    ,    , 114, 76, 145, 93, 90,  60,

        'Avançar peu esquerra + 'Baixar peu esquerre
        MOVE24  90,  56, 143, 122, 114,  60,  80,  40,  80,    ,    ,
        , 105, 40, 80,    ,    ,    , 113, 80, 145,  90,  90,  60,
        MOVE24  90,  46, 163, 112, 114,  60,  80,  40,  80,    ,    ,
        105,  40,  80,    ,    ,    , 112,  80, 145,  90,  90,  60,
```

```
        'Inclinar-se a l'esquerra + Aixeca peu dret
        SPEED 5
        MOVE24 100,  66, 141, 113, 100, 100,  90,  40,  80,    ,    ,
        , 100, 40,  80,    ,    ,    , 100,  83, 156,  80, 100, 100,
        MOVE24 113,  78, 142, 105,  90,  60, 100,  40,  80,    ,    ,
        , 100, 40,  80,    ,    ,    ,  90, 102, 136,  85, 114,  60,


        'Avançar peu dret
        SPEED 7
        MOVE24 113,  76, 145,  93,  90,  60, 100,  40,  80,    ,    ,
        , 90,  40,  80,    ,    ,    ,  90, 107, 105, 105, 114,  60,

        'Avançar peu dret + Baixar peu dret
        SPEED 10
        MOVE24 113,  80, 145,  90,  90,  60, 105,  40,  80,    ,    ,
        , 80,  40,  80,    ,    ,    ,  90,  56, 143, 122, 114,  60,
        MOVE24 112,  80, 145,  90,  90,  60, 105,  40,  80,    ,    ,
        , 80,  40,  80,    ,    ,    ,  90,  46, 163, 112, 114,  60,

        'Inclinar-se a la dreta + Aixecar peu esquerre
        SPEED 5
        MOVE24 100,  83, 156,  80, 100, 100, 100,  40,  80,    ,    ,
        , 90,  40,  80,    ,    ,    , 100,  66, 141, 113, 100, 100,
        MOVE24  90, 102, 136,  85, 114,  60, 100,  40,  80,    ,    ,
        , 100, 40, 80,    ,    ,    , 113,  78, 142, 105,  90,  60,

        'Avançar peu esquerre
        SPEED 7
        MOVE24  90, 107, 105, 105, 114,  60,  90,  40,  80,    ,    ,
        , 100,  40, 80,    ,    ,    , 114,  76, 145,  93,  90,  60,

        RETURN
```

En aquesta primera part es parteix de la posició estàndard i es realitzen dos passos (un amb cada peu). Posteriorment es porta el robot a una posició intermèdia que permetrà continuar caminant si es desitja. Deixant el robot en aquesta posició intermèdia permet realitzar el moviment de caminar de manera més continua i real.

```
caminar:

        SPEED 10
        'Avançar peu esquerra + 'Baixar peu esquerre
        MOVE24  90,  56, 143, 122, 114,  60,  80,  40,  80,    ,    ,
        , 105, 40,  80,    ,    ,    , 113,  80, 145,  90,  90,  60,
        MOVE24  90,  46, 163, 112, 114,  60,  80,  40,  80,    ,    ,
        , 105, 40, 80,    ,    ,    , 112,  80, 145,  90,  90,  60,

        'Inclinar-se a l'esquerra + Aixeca peu dret
        SPEED 5
        MOVE24 100,  66, 141, 113, 100, 100,  90,  40,  80,    ,    ,
        , 100,  40,  80,    ,    ,    , 100,  83, 156,  80, 100, 100,
        MOVE24 113,  78, 142, 105,  90,  60, 100,  40,  80,    ,    ,
        , 100,  40,  80,    ,    ,    ,  90, 102, 136,  85, 114,  60,
```

```
      'Avançar peu dret
      SPEED 7
      MOVE24 113,  76, 145,  93,  90,  60, 100,  40,  80,    ,    ,
      ,  90,  40,  80,    ,    ,    ,  90, 107, 105, 105, 114,  60,

      'Avançar peu dret + Baixar peu dret
      SPEED 10
      MOVE24 113,  80, 145,  90,  90,  60, 105,  40,  80,    ,    ,
      ,  80,  40,  80,    ,    ,    ,  90,  56, 143, 122, 114,  60,
      MOVE24 112,  80, 145,  90,  90,  60, 105,  40,  80,    ,    ,
      ,  80,  40,  80,    ,    ,    ,  90,  46, 163, 112, 114,  60,

      'Inclinar-se a la dreta + Aixecar peu esquerre
      SPEED 5
      MOVE24 100,  83, 156,  80, 100, 100, 100,  40,  80,    ,    ,
      ,  90,  40,  80,    ,    ,    , 100,  66, 141, 113, 100, 100,
      MOVE24  90, 102, 136,  85, 114,  60, 100,  40,  80,    ,    ,
      , 100,  40,  80,    ,    ,    , 113,  78, 142, 105,  90,  60,

      'Avançar peu esquerre
      SPEED 7
      MOVE24  90, 107, 105, 105, 114,  60,  90,  40,  80,    ,    ,
      , 100,  40,  80,    ,    ,    , 114,  76, 145,  93,  90,  60,

      RETURN
```

A la segona part es parteix de la posició intermèdia, es realitzen dos passos consecutius tal i com es feia en el primer pas i es retorna a la posició intermèdia.

```
retorn_inici:

      'Baixa peu esquerre fins alçada peu dret
      SPEED 5
      MOVE24  85,  71, 152,  91, 112,  60, 100,  40,  80,    ,    ,
      , 100,  40,  80,    ,    ,    , 112,  76, 145,  93,  92,  60,

      'standard_pose
      MOVE G6A,100, 76, 145, 93, 100, 100
      MOVE G6D,100, 76, 145, 93, 100, 100
      MOVE G6B,100, 30, 80, 100, 100, 100
      MOVE G6C,100, 30, 80, 100, 100, 100
      WAIT

      RETURN
```

Finalment en el aquesta darrera part es porta el robot de la posició intermèdia fins a la posició estàndard de seguretat. En cas de només voler realitzar un pas els moviments a realitzar simplement serien el primer pas inicial + retorn a l'inici

```
      '================================================
      '================================================
girar_esquerra:
      SPEED 6
      MOVE G6D,  85,  71, 152,  91, 112,  60
      MOVE G6A, 112,  76, 145,  93,  92,  60
```

```
      MOVE G6C, 100,  40,  80,    ,    ,    ,
      MOVE G6B, 100,  40,  80,    ,    ,    ,
      WAIT

      SPEED 9
      MOVE G6A, 113,  75, 145,  97,  93,  60
      MOVE G6D,  90,  50, 157, 115, 112,  60
      MOVE G6B, 105,  40,  70,    ,    ,    ,
      MOVE G6C,  90,  40,  70,    ,    ,    ,
      WAIT

      MOVE G6A, 108,  78, 145,  98,  93,  60
      MOVE G6D,  95,  43, 169, 110, 110,  60
      MOVE G6B, 105,  40,  70,    ,    ,    ,
      MOVE G6C,  80,  40,  70,    ,    ,    ,
      WAIT

      RETURN

    '================================================
    '================================================
lateral_dreta:

      SPEED 5
      GOSUB right_shift1
      SPEED 9
      GOSUB right_shift2

      GOSUB right_shift3
      GOSUB right_shift4

      SPEED 9
      GOSUB right_shift5
      GOSUB right_shift6

      DELAY 200
      GOSUB standard_pose
      DELAY 500

right_shift1:
      MOVE G6D,  85,  71, 152,  91, 112, 60
      MOVE G6A, 112,  76, 145,  93,  92, 60
      MOVE G6C, 100,  40,  80,  ,  ,  ,
      MOVE G6B, 100,  40,  80,  ,  ,  ,
      WAIT
      RETURN

right_shift2:
      MOVE G6A,110,  92, 124,  97,  93,  70
      MOVE G6D, 76,  72, 160,  82, 128,  70
      MOVE G6B,100,  35,  90, , , ,
      MOVE G6C,100,  35,  90, , , ,
      WAIT
      RETURN

right_shift3:
      MOVE G6A, 93,  76, 145,  94, 109, 100
      MOVE G6D, 93,  76, 145,  94, 109, 100
```

```
      MOVE G6B,100,  35,  90, , , ,
      MOVE G6C,100,  35,  90, , , ,
      WAIT
      RETURN

right_shift4:
      MOVE G6D,110,  92, 124,  97,  93,  70
      MOVE G6A, 76,  72, 160,  82, 128,  70
      MOVE G6B,100,  35,  90, , , ,
      MOVE G6C,100,  35,  90, , , ,
      WAIT
      RETURN

right_shift5:
      MOVE G6A, 86,  83, 135,  97, 114,  60
      MOVE G6D,113,  78, 145,  93,  93,  60
      MOVE G6B, 90,  40,  80, , , ,
      MOVE G6C,100,  40,  80, , , ,
      WAIT
      RETURN

right_shift6:
      MOVE G6A, 85,  71, 152,  91, 112, 60
      MOVE G6D,112,  76, 145,  93,  92, 60
      MOVE G6B,100,  40,  80, , , ,
      MOVE G6C,100,  40,  80, , , ,
      WAIT
      RETURN
```

S'inclouen els diferents subprogrames i posicions als que cal portar els servos per poder moure's lateralment i per a poder fer girs (cal recordar que per a girar es fa servir la fricció amb el terra degut a la absència d'un grau de llibertat). La velocitat i separació entre els diferents moviments ve donat pel programa de demostració tal i com s'ha dit anteriorment.

B.2.7. Exercici 6

Exercici 6. ***Fer que el robot camini indefinidament fins a trobar-se amb un obstacle que serà detectat mitjançant el sensor de ultrasons, moment en que s'aturarà. Posteriorment comprovarà si té un obstacle lateral mitjançant el sensor de infrarojos situat al braç. Si no existeix obstacle lateral al costat dret realitzarà un gir de 90 º cap aquell costat i seguirà caminant. En cas que hi hagi obstacle lateral al costat dret, esperarà una senyal acústica durant 5 segons. Si no arriba la senyal el robot haurà d'aturar-se. Si capta una senyal acústica haurà de realitzar de realitzar un gir de 90 º cap al costat esquerre (el que no té sensor de infrarojos) i seguir caminant.***

```
'Declaració de constants
CONST DIST_MIN_FRONT = 25
CONST DIST_MIN_LAT = 25
CONST MAX_PASSOS = 4
CONST MAX_TEMPS = 135
CONST MAX_GIRS_DRETA = 5
CONST MAX_GIRS_ESQUERRA = 4
```

```
CONST SO_MIN = 70
```

Es declaren les constants i els seus valors consegüents que es vol que tinguin.

```
'Declaració de variables
DIM infraroig AS INTEGER
DIM sons AS BYTE
DIM ultrasons AS INTEGER
DIM comptador AS BYTE
DIM passos AS BYTE
DIM girs_dreta AS BYTE
DIM girs_esquerra AS BYTE
DIM obstacle_front AS BYTE
DIM obstacle_lat AS BYTE
```

Es declaren les variables de tipus byte (8 bits - [0-255]) i de tipus integer (16 bits – [0-65535]) per usar-les posteriorment en el programa.

```
'Configure PTP for using with groups of servos
PTP SETON
'Configure PTP for using with all the servos in the robot
PTP ALLON
```

Activa el funcionament punt a punt de tots els servomotors.

```
'== motor direction setting ====================
DIR G6A,1,0,0,1,0,0
DIR G6B,1,1,1,1,1,1
DIR G6C,0,0,0,0,0,0
DIR G6D,0,1,1,0,1,0
```

Es defineix el sentit de gir dels diferents servomotors. Si hi ha un 1 el servomotor girarà en sentit horari mentre que un 0 farà que el servo giri en sentit antihorari.

La definició de moviments i tractaments pels servos es fa sempre per 24 motors (dels quals realment només estan 16).

Grup 6A: Cama esquerra (5 servomotors)

Grup 6B: Braç esquerre(3 servomotors)

Grup 6C: Braç dret (3 servomotors)

Grup 6D: Cama dreta (5 servomotors).

```
'== motor start position read ==================
GETMOTORSET G6A,1,1,1,1,1,0
GETMOTORSET G6B,1,1,1,0,0,0
```

```
GETMOTORSET G6C,1,1,1,0,0,0
GETMOTORSET G6D,1,1,1,1,1,0
```

Definició de la posició inicial al iniciar el programa. En cas de ser un 1 els motors es quedaran a la posició que estaven, en cas de tenir un 0 es mouran a una posició de seguretat indicada per el controladors.

```
SPEED 5
```

S'assigna a la velocitat el valor 5. Es recomana no utilitza velocitats superiors a 10 per les diferents pràctiques a realitzar

```
'== motor power on ============================
MOTOR G24
GOSUB standard_pose

'================================================
'================================================

'Main procedure
MAIN:


    ' Inicialització
    passos = 0
    obstacle_front = 0
    obstacle_lat = 0
    comptador = 0

    DELAY 3000
```

S'inicialitzen les variables que es faran servir durant el programa i es fa una pausa de 3 segons abans de començar

```
      'Captura de la distancia frontal amb ultrasons
      GOSUB capturar_valor_ultrasons


      'Primera comprovació de distancia i primer pas
      IF ultrasons < DIST_MIN_FRONT THEN
            obstacle_front = 1
      ELSE
            GOSUB primer_pas
      ENDIF


      'Comprovació reiterativa de distancia i caminar
      IF obstacle_front = 0 THEN
            FOR passos = 1 TO MAX_PASSOS
                  IF obstacle_front = 0 THEN
                        GOSUB capturar_valor_ultrasons
                        IF ultrasons < DIST_MIN_FRONT THEN
                              obstacle_front = 1
```

```
                        ELSEIF ultrasons >= DIST_MIN_FRONT THEN
                              GOSUB caminar
                        ENDIF
                  ENDIF
            NEXT passos
            GOSUB retorn_inici
      ENDIF
```

Es capturar el valor del sensor de ultrasons. En cas que es trobi un obstacle frontal a una distància inferior a la mínima (calculada prèviament per tal que el robot pugui fer un pas complet), s'activa la variable obstacle. En cas contrari es farà el primer pas (primera part del moviment de caminar).

Posteriorment si no s'ha trobat obstacle es realitzaran diversos passos consecutius. Abans de fer cada pas, es capturarà el valor del sensor per veure si existeix obstacle. Finalment tant si es para el robot per un obstacle com si acaba de fer tots els passos tornarà a la posició estàndard. La utilització de dos sentències if i una for es degut a que el llenguatge RoboBasic no conté cap sentència while.

```
      ' Moviment final segons situació
      DELAY 2000
      IF obstacle_front = 0 THEN
            GOTO MAIN
      ELSEIF obstacle_front = 1 THEN
            GOSUB moviment_lateral
      ENDIF
```

Si no hi ha cap obstacle frontal es reiniciarà el programa. Si existeix un obstacle passarà al subprograma moviment_lateral.

```
moviment_lateral:

      'Comprovació distancia lateral amb sensor d'infrarojos
      GOSUB capturar_valor_infraroig
      IF infraroig  <= DIST_MIN_LAT THEN
            obstacle_lat = 1
      ENDIF

      'Actuar segons si existeix obstacle lateral o no

      IF obstacle_lat = 1 THEN
            MUSIC "7D"
            GOTO captar_so
      ELSEIF obstacle_lat = 0 THEN
            GOSUB gir_90_dreta
            GOSUB standard_pose
            GOTO MAIN
      ENDIF
```

De manera semblant al realitzat amb l'obstacle lateral es comprova si hi ha un obstacle lateral mitjançant el sensor de infrarojos. Si hi ha obstacle lateral es produirà un so i es passarà al subprograma captar_so, si no hi ha el robot girarà 90º cap a la dreta i es tornarà a iniciar el programa.

```
captar_so:

      'En cas d'obstacle lateral esperar una ordre mitjançant un xiulet
      (durant un cert temps després del senyal emes) i actuarà segons la
      rebi o no
      comptador = comptador + 1
      sons = AD(4)
      IF comptador <= MAX_TEMPS THEN
            IF sons > SO_MIN THEN
                  DELAY 1000
                  GOSUB gir_90_esquerra
                  DELAY 1000
                  GOSUB standard_pose
                  DELAY 1000
                  GOTO MAIN
            ELSEIF sons <= SO_MIN THEN
                  GOTO captar_so
            ENDIF
      ENDIF

      'En cas de passar els 5 o 6 segons el robot s'aturarà.

      SPEED 10
      MOVE G6A, 100, 151,  23, 140, 101, 100
      MOVE G6D, 100, 151,  23, 140, 101, 100
      MOVE G6B, 100,  30,  80, 100, 100, 100
      MOVE G6C, 100,  30,  80, 100, 100, 100
      WAIT
      '== MOTOR power off  ============================
      MOTOROFF G24

      END
```

Si s'ha trobat obstacle frontal i lateral al costat dret com no es tenen dos sensors de infrarojos es farà servir el sensor de so per tal que l'usuari sigui qui ho indiqui. Per tal d'esperar durant 5 o 6 segons, com no es té un comptador de temps en llenguatge RoboBasic, es realitza un bucle que va sumant 1 a la variable comptador. Aquest procés s'anirà repetint fins que es capti un so mitjançant el sensor o el comptador arribi al MAX_TEMPS (el valor d'aquesta constant s'haurà calculat prèviament de forma que el temps de còmput de la controladora per processar les diferents sentències dins el bucle coincideixi amb el valor que es vol que el robot estigui esperant captar el so).

Si es capta un so per sobre del valor límit el robot girarà 90 º cap a l'esquerra i tornarà a iniciar el programa (de forma que seguirà caminant). En canvi, si passen els 5 o 6 segons sense rebre cap so, el robot s'ajupirà, s'apagaran tots els servomotor i el programa finalitzarà.

```
      '================================================
      '================================================
standard_pose:
      SPEED 5
      MOVE G6A,100, 76, 145, 93, 100, 100
      MOVE G6D,100, 76, 145, 93, 100, 100
      MOVE G6B,100, 30, 80, 100, 100, 100
```

```
      MOVE G6C,100, 30, 80, 100, 100, 100
      WAIT
      RETURN
```

Es defineixen els valors de la posició estàndard (posició inicial i de seguretat).

```
    '================================================
    '================================================
capturar_valor_infraroig:

      'Captar valor, fer la conversió a cm i en cas de no trobar-se en el
      rang de treball tornem a captar el valor
      infraroig = AD (6)
      IF infraroig < 4 THEN infraroig = 4    'min
      infraroig = infraroig - 3
      infraroig = 6787 / infraroig

      IF infraroig < 10 OR infraroig > 150 THEN
      GOTO  capturar_valor_infraroig

      WAIT
      RETURN
```

Es fa servir el subprograma explicat anteriorment per a captar el valor del sensor d'infraroigs

```
      '================================================
      '================================================
      capturar_valor_ultrasons:

      ultrasons = AD(2)
      ultrasons = ultrasons * 13
      ultrasons = ultrasons / 10

      WAIT
      RETURN
```

Es capta el valor del sensor de ultrasons i es transforma a cm fent servir les comandes explicades.

```
          '================================================
      '================================================

primer_pas:

      'Inclinar-se a la dreta
       SPEED 5
      MOVE24 85, 71, 152, 91, 112, 60, 100, 40,  80,    ,    ,    ,
      100,  40,  80,    ,    ,    , 112,  76, 145,  93,  92,  60,

      'Aixecar peu esquerre
      SPEED 10
      MOVE24  90, 107, 105, 105, 114,  60,  90,  40,  80,    ,    ,
      , 100,  40,  80,    ,    ,    , 114, 76, 145, 93, 90,  60,

      'Avançar peu esquerra + 'Baixar peu esquerre
      MOVE24  90,  56, 143, 122, 114,  60,  80,  40,  80,    ,    ,
```

```
      , 105, 40, 80,    ,    ,    , 113, 80, 145,  90,  90,  60,
      MOVE24  90,  46, 163, 112, 114,  60,  80,  40,  80,    ,    ,
      105,  40,  80,    ,    ,    , 112,  80, 145,  90,  90,  60,

      'Inclinar-se a l'esquerra + Aixeca peu dret
      SPEED 5
      MOVE24 100,  66, 141, 113, 100, 100,  90,  40,  80,    ,    ,
      , 100, 40,  80,    ,    ,    , 100,  83, 156,  80, 100, 100,
      MOVE24 113,  78, 142, 105,  90,  60, 100,  40,  80,    ,    ,
      , 100, 40,  80,    ,    ,    ,  90, 102, 136,  85, 114,  60,


      'Avançar peu dret
      SPEED 7
      MOVE24 113,  76, 145,  93,  90,  60, 100,  40,  80,    ,    ,
      , 90,  40,  80,    ,    ,    ,  90, 107, 105, 105, 114,  60,

      'Avançar peu dret + Baixar peu dret
      SPEED 10
      MOVE24 113,  80, 145,  90,  90,  60, 105,  40,  80,    ,    ,
      , 80,  40,  80,    ,    ,    ,  90,  56, 143, 122, 114,  60,
      MOVE24 112,  80, 145,  90,  90,  60, 105,  40,  80,    ,    ,
      , 80,  40,  80,    ,    ,    ,  90,  46, 163, 112, 114,  60,

      'Inclinar-se a la dreta + Aixecar peu esquerre
      SPEED 5
      MOVE24 100,  83, 156,  80, 100, 100, 100,  40,  80,    ,    ,
      , 90,  40,  80,    ,    ,    , 100,  66, 141, 113, 100, 100,
      MOVE24  90, 102, 136,  85, 114,  60, 100,  40,  80,    ,    ,
      , 100, 40, 80,    ,    ,    , 113,  78, 142, 105,  90,  60,

      'Avançar peu esquerre
      SPEED 7
      MOVE24  90, 107, 105, 105, 114,  60,  90,  40,  80,    ,    ,
      , 100,  40, 80,    ,    ,    , 114,  76, 145,  93,  90,  60,

      RETURN
```

En aquesta primera part es parteix de la posició estàndard i es realitzen dos passos (un amb cada peu). Posteriorment es porta el robot a una posició intermèdia que permetrà continuar caminant si es desitja. Deixant el robot en aquesta posició intermèdia permet realitzar el moviment de caminar de manera més continua i real.

```
caminar:

      SPEED 10
      'Avançar peu esquerra + 'Baixar peu esquerre
      MOVE24  90,  56, 143, 122, 114,  60,  80,  40,  80,    ,    ,
      , 105, 40,  80,    ,    ,    , 113,  80, 145,  90,  90,  60,
      MOVE24  90,  46, 163, 112, 114,  60,  80,  40,  80,    ,    ,
      , 105, 40, 80,    ,    ,    , 112,  80, 145,  90,  90,  60,

      'Inclinar-se a l'esquerra + Aixeca peu dret
      SPEED 5
      MOVE24 100,  66, 141, 113, 100, 100,  90,  40,  80,    ,    ,
      , 100,  40,  80,    ,    ,    , 100,  83, 156,  80, 100, 100,
```

```
        MOVE24 113,  78, 142, 105,  90,  60, 100,  40,  80,    ,    ,
        , 100,  40,  80,    ,    ,    ,  90, 102, 136,  85, 114,  60,

        'Avançar peu dret
        SPEED 7
        MOVE24 113,  76, 145,  93,  90,  60, 100,  40,  80,    ,    ,
        ,  90,  40,  80,    ,    ,    ,  90, 107, 105, 105, 114,  60,

        'Avançar peu dret + Baixar peu dret
        SPEED 10
        MOVE24 113,  80, 145,  90,  90,  60, 105,  40,  80,    ,    ,
        ,  80,  40,  80,    ,    ,    ,  90,  56, 143, 122, 114,  60,
        MOVE24 112,  80, 145,  90,  90,  60, 105,  40,  80,    ,    ,
        ,  80,  40,  80,    ,    ,    ,  90,  46, 163, 112, 114,  60,

        'Inclinar-se a la dreta + Aixecar peu esquerre
        SPEED 5
        MOVE24 100,  83, 156,  80, 100, 100, 100,  40,  80,    ,    ,
        ,  90,  40,  80,    ,    ,    , 100,  66, 141, 113, 100, 100,
        MOVE24  90, 102, 136,  85, 114,  60, 100,  40,  80,    ,    ,
        , 100,  40,  80,    ,    ,    , 113,  78, 142, 105,  90,  60,

        'Avançar peu esquerre
        SPEED 7
        MOVE24  90, 107, 105, 105, 114,  60,  90,  40,  80,    ,    ,
        , 100,  40,  80,    ,    ,    , 114,  76, 145,  93,  90,  60,

        RETURN
```

A la segona part es parteix de la posició intermèdia, es realitzen dos passos consecutius tal i com es feia en el primer pas i es retorna a la posició intermèdia.

```
retorn_inici:

        'Baixa peu esquerre fins alçada peu dret
        SPEED 5
        MOVE24  85,  71, 152,  91, 112,  60, 100,  40,  80,    ,    ,
        , 100,  40,  80,    ,    ,    , 112,  76, 145,  93,  92,  60,

        'standard_pose
        MOVE G6A,100, 76, 145, 93, 100, 100
        MOVE G6D,100, 76, 145, 93, 100, 100
        MOVE G6B,100, 30, 80, 100, 100, 100
        MOVE G6C,100, 30, 80, 100, 100, 100
        WAIT

        RETURN
```

Finalment en el aquesta darrera part es porta el robot de la posició intermèdia fins a la posició estàndard de seguretat. En cas de només voler realitzar un pas els moviments a realitzar simplement serien el primer pas inicial + retorn a l'inici

```
        '================================================
        '================================================
```

```
gir_90_dreta:

      girs_dreta = 1
      DELAY 4000
      FOR girs_dreta = 1 TO MAX_GIRS_DRETA
            GOSUB turn_right
            GOSUB standard_pose
            DELAY 1000
      NEXT girs_dreta

      RETURN

turn_right:
      SPEED 6

      MOVE G6A,  85,  71, 152,  91, 112,  60
      MOVE G6D, 112,  76, 145,  93,  92,  60
      MOVE G6B, 100,  40,  80,    ,    ,    ,
      MOVE G6C, 100,  40,  80,    ,    ,    ,
      WAIT

      SPEED 9

      MOVE G6D, 113,  75, 145,  97,  93,  60
      MOVE G6A,  90,  50, 157, 115, 112,  60
      MOVE G6C, 105,  40,  70,    ,    ,    ,
      MOVE G6B,  90,  40,  70,    ,    ,    ,
      WAIT

      MOVE G6D, 108,  78, 145,  98,  93,  60
      MOVE G6A,  95,  43, 169, 110, 110,  60
      MOVE G6C, 105,  40,  70,    ,    ,    ,
      MOVE G6B,  80,  40,  70,    ,    ,    ,
      WAIT

      RETURN


    '================================================
    '================================================

gir_90_esquerra:
      girs_esquerra = 1
      DELAY 4000
      FOR girs_esquerra = 1 TO MAX_GIRS_ESQUERRA
            GOSUB turn_left
            GOSUB standard_pose
            DELAY 1000
      NEXT girs_esquerra

      RETURN

turn_left:
      SPEED 6

      MOVE G6D,  85,  71, 152,  91, 112,  60
      MOVE G6A, 112,  76, 145,  93,  92,  60
      MOVE G6C, 100,  40,  80,    ,    ,    ,
```

```
MOVE G6B, 100,  40,  80,    ,    ,    ,
WAIT

SPEED 9

MOVE G6A, 113,  75, 145,  97,  93,  60
MOVE G6D,  90,  50, 157, 115, 112,  60
MOVE G6B, 105,  40,  70,    ,    ,    ,
MOVE G6C,  90,  40,  70,    ,    ,    ,
WAIT

MOVE G6A, 108,  78, 145,  98,  93,  60
MOVE G6D,  95,  43, 169, 110, 110,  60
MOVE G6B, 105,  40,  70,    ,    ,    ,
MOVE G6C,  80,  40,  70,    ,    ,    ,
WAIT

RETURN
```

S'inclouen els diferents subprogrames i posicions als que cal portar els servos per poder fer girs (cal recordar que per a girar es fa servir la fricció amb el terra degut a la absència d'un grau de llibertat). La velocitat i separació entre els diferents moviments ve donat pel programa de demostració tal i com s'ha dit anteriorment.

B.2.8. Exercici 7 (examen)

Exercici 7 (examen). ***S'ha de preparar el robot per a un combat. Al començar el robot haurà de fer una reverència. Posteriorment es posarà en posició defensiva (ajagut parcialment i amb els dos braços davant el pit com a protecció, sense tapar el sensor de ultrasons). Restarà en aquesta posició i anirà comprovant amb el sensor de ultrasons i amb el de infrarojos si l'enemic s'acosta frontalment o lateralment.***
Quan l'enemic sigui detectat a una distància inferior a 15 cm realitzarà puntades de peu en el cas frontal o bé cop de colze en el cas lateral. Una vegada realitzats els moviments ofensius, esperarà durant 5 segons el xiulet confirmant la derrota del contrincant, moment en el que realitzarà un moviment de celebració. En cas de no captar el xiulet tornarà a la posició defensiva de nou.
Caldrà en tot moment comprovar l'estabilitat del robot després de realitzar els moviments i en cas de caiguda s'aixecarà però es considerarà que el robot ha perdut i s'apagarà.

```
'Declaració de constants
CONST DIST_MIN_LAT = 25
CONST DIST_MIN_FRONT = 20
CONST MAX_TEMPS = 135
CONST SO_MIN = 70
CONST MIN_X = 120
CONST MAX_X = 165
CONST MIN_Y = 115
CONST MAX_Y = 165
```

Es declaren les constants i els seus valors consegüents que es vol que tinguin.

```
'Declaració de variables
DIM infraroig AS INTEGER
DIM ultrasons AS INTEGER
DIM sons AS BYTE
DIM eix_x AS BYTE
DIM eix_y AS BYTE
DIM caiguda AS BYTE
DIM mov_combat AS BYTE
DIM comptador AS BYTE
```

Es declaren les variables de tipus byte (8 bits - [0-255]) i de tipus integer (16 bits – [0-65535]) per usar-les posteriorment en el programa.

```
'Configure PTP for using with groups of servos
PTP SETON
'Configure PTP for using with all the servos in the robot
PTP ALLON
```

Activa el funcionament punt a punt de tots els servomotors.

```
'== motor direction setting ====================
DIR G6A,1,0,0,1,0,0
DIR G6B,1,1,1,1,1,1
DIR G6C,0,0,0,0,0,0
DIR G6D,0,1,1,0,1,0
```

Es defineix el sentit de gir dels diferents servomotors. Si hi ha un 1 el servomotor girarà en sentit horari mentre que un 0 farà que el servo giri en sentit antihorari.

La definició de moviments i tractaments pels servos es fa sempre per 24 motors (dels quals realment només estan 16).

Grup 6A: Cama esquerra (5 servomotors)

Grup 6B: Braç esquerre(3 servomotors)

Grup 6C: Braç dret (3 servomotors)

Grup 6D: Cama dreta (5 servomotors).

```
'== motor start position read ==================
GETMOTORSET G6A,1,1,1,1,1,0
GETMOTORSET G6B,1,1,1,0,0,0
GETMOTORSET G6C,1,1,1,0,0,0
GETMOTORSET G6D,1,1,1,1,1,0
```

Definició de la posició inicial al iniciar el programa. En cas de ser un 1 els motors es quedaran a la posició que estaven, en cas de tenir un 0 es mouran a una posició de seguretat indicada per el controladors.

```
SPEED 5
```

S'assigna a la velocitat el valor 5. Es recomana no utilitza velocitats superiors a 10 per les diferents pràctiques a realitzar

```
'== motor power on ============================
MOTOR G24
GOSUB standard_pose

'================================================
'================================================

INICI:

      'Inicialització de variables
      caiguda = 0
      mov_combat = 0


      'Reverencia + Posició defensiva
      DELAY 2500
      GOSUB reverencia
      DELAY 1000
      GOSUB standard_pose
      DELAY 500
      GOSUB mov_defensiu

      'Un cop realitzats els moviments inicials, es passa al mode combat
      GOTO COMBAT
```

El mode INICI del programa consta de la inicialització de les variables, una reverència i el moviment que passa de la posició estàndard a la posició defensiva. Posteriorment es passa a la segona part.

```
COMBAT:

      'Posició defensiva de combat a l'espera d'esdeveniments

      GOSUB pos_defensiva
      DELAY 1000

      'Busquem enemics i ataquem en cas de trobar-ne un (primer
      lateralment i després frontalment)
      GOSUB capturar_valor_infraroig
      IF infraroig < DIST_MIN_LAT THEN
            GOSUB standard_pose
            DELAY 500
            GOSUB atac_lateral
            DELAY 1000
            GOSUB standard_pose
            mov_combat = 1
```

```
      ENDIF
      GOSUB comprovar_caiguda

      GOSUB capturar_valor_ultrasons
      IF ultrasons < DIST_MIN_FRONT THEN
            GOSUB standard_pose
            DELAY 500
            GOSUB atac_frontal
            DELAY 1000
            GOSUB standard_pose
            mov_combat = 1
      ENDIF
      GOSUB comprovar_caiguda
```

En aquesta mode del programa primer es posa el robot en la posició defensiva. Posteriorment es comprova si existeix algun enemic primer lateralment i després frontalment mitjançant els sensors de infraroigs i ultrasons, respectivament. En cas que existeix algun enemic, es realitza un atac cap a la direcció que pertoqui, s'activa la variable mov_combat i es comprova si ha caigut ja sigui per un impacte enemic o en l'execució del mateix.

```
'Nomes es passa al mode comprovar victòria en cas d'haver fet algun
      moviment ofensiu, sinó restarà a la posició defensiva
      IF mov_combat = 1 THEN
            'Aquí cal inicialitzar la variable comptador a 0 ja que cada
            Vegada que comprovi victòria, haurà de comptar el temps
            comptador = 0
            GOTO COMPROVAR_VICTORIA
      ELSEIF mov_combat = 0 THEN
            GOTO COMBAT
      ENDIF
```

Per acabar es comprova si s'ha realitzat algun moviment de combat. En cas afirmatiu s'inicia la variable que es fa servir per comptar el temps i es passa al mode que controla la victòria; sinó, es reiniciarà el mode COMBAT del programa.

```
COMPROVAR_VICTORIA:

      'Espera durant uns 5 segons (el temps que triga a comptador a
      arribar al valor MAX_TEMPS sumant 1 cada vegada que realitza el
      cicle). Hi ha 3 casos:
      '1- Rep un xiulet i passa al mode VICTORIA
      '2- No rep res i torna a realitzar el cicle COMPROVAR_VICTORIA
      '3- Han passat els 5 segons i torna el mode COMBAT

      comptador = comptador + 1
      sons = AD(4)
      IF comptador <= MAX_TEMPS THEN
            IF sons > SO_MIN THEN
                  GOTO VICTORIA
            ELSEIF sons <= SO_MIN THEN
                  GOTO COMPROVAR_VICTORIA
            ENDIF
      ENDIF
```

```
        GOTO COMBAT
```

Per tal de comprovar la victòria es farà servir el sensor de so que captarà el xiulet de l'àrbitre en cas de produir-se. Per tal d'esperar durant 5 o 6 segons, com no es té un comptador de temps en llenguatge RoboBasic, es realitza un bucle que va sumant 1 a la variable comptador. Aquest procés s'anirà repetint fins que es capti un so mitjançant el sensor o el comptador arribi al MAX_TEMPS (el valor d'aquesta constant s'haurà calculat prèviament de forma que el temps de còmput de la controladora per processar les diferents sentències dins el bucle coincideixi amb el valor que es vol que el robot estigui esperant captar el so).

Si es capta el xiulet es passarà al mode victòria. En canvi, si passen els 5 o 6 segons sense rebre cap so, el robot tornarà al mode combat.

```
VICTORIA:

        GOSUB moviment_victòria
        DELAY 1000
        GOSUB standard_pose

        END
```

Finalment en cas d'haver sentit el xiulet es realitzarà el moviment de victòria, es tornarà a la posició estàndard i es finalitzarà el programa.

```
        '================================================
        '================================================
standard_pose:
        SPEED 5
        MOVE G6A,100, 76, 145, 93, 100, 100
        MOVE G6D,100, 76, 145, 93, 100, 100
        MOVE G6B,100, 30, 80, 100, 100, 100
        MOVE G6C,100, 30, 80, 100, 100, 100
        WAIT
        RETURN
```

Es defineixen els valors de la posició estàndard (posició inicial i de seguretat).

```
        '================================================
        '================================================
reverencia:

        MOVE G6A, 100,  58, 135, 160, 100, 100
        MOVE G6D, 100,  58, 135, 160, 100, 100
        MOVE G6B, 100,  30,  80,  ,  ,  ,
        MOVE G6C, 100,  30,  80,  ,  ,  ,
        WAIT
        RETURN

        '================================================
        '================================================
mov_defensiu:

        SPEED 6

        MOVE G6A,  85,  71, 152,  91, 112,  60
```

```
MOVE G6D, 112,  76, 145,  93,  92,  60
MOVE G6B, 100,  40,  80,    ,    ,    ,
MOVE G6C, 100,  40,  80,    ,    ,    ,
WAIT

SPEED 9

MOVE G6D, 113,  75, 145,  97,  93,  60
MOVE G6A,  90,  50, 157, 115, 112,  60
MOVE G6C, 105,  40,  70,    ,    ,    ,
MOVE G6B,  90,  40,  70,    ,    ,    ,
WAIT

MOVE G6D, 108,  78, 145,  98,  93,  60
MOVE G6A,  95,  43, 169, 110, 110,  60
MOVE G6C, 105,  40,  70,    ,    ,    ,
MOVE G6B,  80,  40,  70,    ,    ,    ,
WAIT



SPEED 7

MOVE G6A,100, 76, 145, 93, 100, 100
MOVE G6D,100, 76, 145, 93, 100, 100
MOVE G6B,100, 30, 80, 100, 100, 100
MOVE G6C,100, 30, 80, 100, 100, 100
WAIT

SPEED 6

MOVE G6A,  85,  71, 152,  91, 112,  60
MOVE G6D, 112,  76, 145,  93,  92,  60
MOVE G6B, 100,  40,  80,    ,    ,    ,
MOVE G6C, 100,  40,  80,    ,    ,    ,
WAIT

SPEED 9

MOVE G6D, 113,  75, 145,  97,  93,  60
MOVE G6A,  90,  50, 157, 115, 112,  60
MOVE G6C, 105,  40,  70,    ,    ,    ,
MOVE G6B,  90,  40,  70,    ,    ,    ,
WAIT

MOVE G6D, 108,  78, 145,  98,  93,  60
MOVE G6A,  95,  43, 169, 110, 110,  60
MOVE G6C, 105,  40,  70,    ,    ,    ,
MOVE G6B,  80,  40,  70,    ,    ,    ,
WAIT

SPEED 7

MOVE G6A,100, 76, 145, 93, 100, 100
MOVE G6D,100, 76, 145, 93, 100, 100
MOVE G6B,100, 30, 80, 100, 100, 100
MOVE G6C,100, 30, 80, 100, 100, 100
WAIT
```

```
      DELAY 500

      MOVE G6A, 101,  73, 102, 136,  99,
      MOVE G6D, 103, 130, 100,  81,  96,
      MOVE G6B, 143,  37,  62,  ,  ,
      MOVE G6C, 188,  28,  52,  ,  ,
      WAIT

      MOVE G6A, 101,  73, 102, 136,  99,
      MOVE G6D, 103, 130, 100,  81,  96,
      MOVE G6B, 159,  16,  52,  ,  ,
      MOVE G6C, 188,  28,  52,  ,  ,
      WAIT

      RETURN

      '================================================
      '================================================
pos_defensiva:

      MOVE G6A, 101,  73, 102, 136,  99,
      MOVE G6D, 103, 130, 100,  81,  96,
      MOVE G6B, 159,  16,  52,  ,  ,
      MOVE G6C, 188,  28,  52,  ,  ,
      WAIT

      RETURN
```

Per cadascun dels subprogrames(reverència, moviment defensiu i posició defensiva) s'introdueixen les posicions a les que cal portar els servos mitjançant l'eina Catch & Play. En aquest cas s'han introduït en grups de 6 servomotors. S'esperarà a que els servomotors finalitzin el moviment i es retornarà al programa principal.

```
      '================================================
      '================================================
capturar_valor_infraroig:

      infraroig = AD (6)
      IF infraroig < 4 THEN infraroig = 4     'min
      infraroig = infraroig - 3
      infraroig = 6787 / infraroig

      IF infraroig < 10 OR infraroig > 150 THEN
            GOTO capturar_valor_infraroig

      WAIT
      RETURN
```

Es fa servir el subprograma explicat anteriorment per a captar el valor del sensor d'infraroigs

```
      '================================================
      '================================================
capturar_valor_ultrasons:
```

```
      ultrasons = AD(2)
      ultrasons = ultrasons * 13
      ultrasons = ultrasons / 10

      WAIT
      RETURN
```

Es capta el valor del sensor de ultrasons i es transforma a cm fent servir les comandes explicades.

```
      '================================================
      '================================================
comprovar_caiguda:

    DELAY 500
    eix_x = AD(1)
    IF eix_x < MIN_X THEN
        GOSUB aixecar_esquerra
        DELAY 500
        GOSUB standard_pose
        caiguda = 1
    ELSEIF eix_x > MAX_X THEN
        GOSUB aixecar_dreta
        DELAY 500
        GOSUB standard_pose
        caiguda = 1
    ENDIF

    DELAY 500
    eix_y = AD(0)
    IF eix_y < MIN_Y THEN
        GOSUB aixecar_enrere
        DELAY 500
        GOSUB standard_pose
        caiguda = 1
    ELSEIF eix_y > MAX_Y THEN
        GOSUB aixecar_endav
        DELAY 500
        GOSUB standard_pose
        caiguda = 1
    ENDIF

    IF caiguda = 0 THEN
        RETURN
    ELSEIF caiguda = 1 THEN
        GOTO apagar_caiguda
    ENDIF
```

Es comprova la caiguda primer a l'eix x i després a l'eix y mitjançant el valor captat per l'acceleròmetre. En cas de caiguda s'executarà el subprograma per aixecar que correspongui i s'activarà la variable de caiguda. Finalment es comprova l'estat de la variable caiguda, en cas de ser 0 es tornarà al programa principal mentre que si està activa es passarà al subprograma que apagarà el robot.

```
      '================================================
      '================================================
```

```
aixecar_dreta:
      SPEED 10

      MOVE G6A, 100,  76, 145, 180, 100,
      MOVE G6D, 100,  76, 145,  57, 100,
      MOVE G6B, 100,  30,  80,  ,  ,
      MOVE G6C, 100,  30,  80,  ,  ,

      MOVE G6A,100, 76, 145, 93, 100, 100
      MOVE G6D,100, 76, 145, 93, 100, 100
      MOVE G6B,100, 30, 80, 100, 100, 100
      MOVE G6C,100, 30, 80, 100, 100, 100

      WAIT
      RETURN

aixecar_esquerra:
      SPEED 10
      MOVE G6A, 100,  76, 145,  93, 100,
      MOVE G6D, 100,  76, 145,  14, 100,
      MOVE G6B, 100,  30,  80,  ,  ,
      MOVE G6C, 100,  30,  80,  ,  ,

      MOVE G6A,100, 76, 145, 93, 100, 100
      MOVE G6D,100, 76, 145, 93, 100, 100
      MOVE G6B,100, 30, 80, 100, 100, 100
      MOVE G6C,100, 30, 80, 100, 100, 100

      WAIT
      RETURN
```

En cas de caiguda lateral, degut a que els braços no tenen suficient força per aixecar el robot, el que es fa es moure el braç contrari endavant o enrere de forma que el robot es desequilibri i passi d'una posició de caiguda lateral a caiguda frontal o posterior, respectivament. Posteriorment tal i com s'ha posat l'ordre de les comandes s'aixecarà amb els subprogrames consegüents.

```
      '================================================
      '================================================
aixecar_endav:

      SPEED 10

      MOVE G6A,100, 130, 120,  80, 110, 100
      MOVE G6D,100, 130, 120,  80, 110, 100
      MOVE G6B,150, 160,  10, 100, 100, 100
      MOVE G6C,150, 160,  10, 100, 100, 100
      WAIT

      MOVE G6A, 80, 155,  85, 150, 150, 100
      MOVE G6D, 80, 155,  85, 150, 150, 100
      MOVE G6B,185,  40, 60,  100, 100, 100
      MOVE G6C,185,  40, 60,  100, 100, 100
      WAIT

      MOVE G6A, 75, 165,  55, 165, 155, 100
```

```
      MOVE G6D, 75, 165,  55, 165, 155, 100
      MOVE G6B,185,  10, 100, 100, 100, 100
      MOVE G6C,185,  10, 100, 100, 100, 100
      WAIT

      MOVE G6A, 60, 165,  30, 165, 155, 100
      MOVE G6B,170,  10, 100, 100, 100, 100
      MOVE G6C,170,  10, 100, 100, 100, 100
      WAIT

      MOVE G6A, 60, 165,  25, 160, 145, 100
      MOVE G6D, 60, 165,  25, 160, 145, 100
      MOVE G6B,150,  60,  90, 100, 100, 100
      MOVE G6C,150,  60,  90, 100, 100, 100
      WAIT

      MOVE G6A,100, 155,  25, 140, 100, 100
      MOVE G6D,100, 155,  25, 140, 100, 100
      MOVE G6B,130,  50,  85, 100, 100, 100
      MOVE G6C,130,  50,  85, 100, 100, 100
      WAIT

      RETURN

aixecar_enrere:

      SPEED 10

      MOVE G6A,100,  10, 100, 115, 100, 100
      MOVE G6D,100,  10, 100, 115, 100, 100
      MOVE G6B,100, 130,  10, 100, 100, 100
      MOVE G6C,100, 130,  10, 100, 100, 100
      WAIT

      MOVE G6A,100, 10,  83, 140, 100, 100
      MOVE G6D,100, 10,  83, 140, 100, 100
      MOVE G6B,20, 130,  10, 100, 100, 100
      MOVE G6C,20, 130,  10, 100, 100, 100
      WAIT

      MOVE G6A,100, 126,  60, 50, 100, 100
      MOVE G6D,100, 126,  60, 50, 100, 100
      MOVE G6B,20,  30,  90, 100, 100, 100
      MOVE G6C,20,  30,  90, 100, 100, 100
      WAIT

      MOVE G6A,100, 165,  70, 15, 100, 100
      MOVE G6D,100, 165,  70, 15, 100, 100
      MOVE G6B, 30,  20,  95,100, 100, 100
      MOVE G6C, 30,  20,  95,100, 100, 100
      WAIT

      MOVE G6A,100, 165,  40, 100, 100, 100
      MOVE G6D,100, 165,  40, 100, 100, 100
      MOVE G6B,110,  70,  50, 100, 100, 100
      MOVE G6C,110,  70,  50, 100, 100, 100
      WAIT
```

```
        RETURN
```

S'inclouen els diferents subprogrames i posicions als que cal portar els servos per poder aixecar-se endavant o enrere. La velocitat i separació entre els diferents moviments ve donat pel programa de demostració tal i com s'ha dit anteriorment.

```
        '================================================
        '================================================
apagar_caiguda:

        SPEED 10
        MOVE G6A, 100, 151,  23, 140, 101, 100
        MOVE G6D, 100, 151,  23, 140, 101, 100
        MOVE G6B, 100,  30,  80, 100, 100, 100
        MOVE G6C, 100,  30,  80, 100, 100, 100
        WAIT

        MOTOROFF G24
        END
```

En cas de caiguda el robot s'ajupirà, s'apagaran els diferents servomotors i finalitzarà el programa.

```
        '================================================
        '================================================
atac_lateral:

        GOSUB right_forward
        GOSUB standard_pose

        RETURN

        '================================================
right_forward:
        SPEED 7
        MOVE G6A, 108,  76, 145,  93, 100, 60
        MOVE G6D,  85,  71, 152,  91, 107, 60
        MOVE G6B,  70,  40,  80,  ,  ,  ,
        MOVE G6C, 130,  40,  80,  ,  ,  ,
        WAIT

        SPEED 10
        HIGHSPEED SETON
        MOVE G6A,  66, 163,  85,  65, 130
        MOVE G6D, 107, 164,  21, 125,  93
        MOVE G6B,  50,  72,  86
        MOVE G6C, 189,  40,  77
        WAIT

        DELAY 1000
        HIGHSPEED SETOFF


        GOSUB sit_pose
        RETURN
```

```
      '================================================
sit_pose:

      SPEED 10
      MOVE G6A,100, 151,  23, 140, 101, 100
      MOVE G6D,100, 151,  23, 140, 101, 100
      MOVE G6B,100,  30,  80, 100, 100, 100
      MOVE G6C,100,  30,  80, 100, 100, 100
      WAIT

      RETURN

      '================================================
      '================================================
atac_frontal:

      GOSUB right_shoot
      GOSUB standard_pose
      DELAY 500
      GOSUB left_shoot
      GOSUB standard_pose
      DELAY 500

      RETURN

      '================================================
right_shoot:
      SPEED 4
      MOVE G6A,112,  56, 180,  79, 104, 100
      MOVE G6D, 70,  56, 180,  79, 102, 100
      MOVE G6B,110,  45,  70, 100, 100, 100
      MOVE G6C, 90,  45,  70, 100, 100, 100
      WAIT

right_shoot1:
      SPEED 6
      MOVE G6A,115,  60, 180,  79,  95,  100
      MOVE G6D, 90,  90, 127,  65, 116,  100
      MOVE G6B, 80,  45,  70, 100, 100, 100
      MOVE G6C,120,  45,  70, 100, 100, 100
      WAIT

      SPEED 15
      HIGHSPEED SETON

right_shoot2:
      MOVE G6A,115,  52, 180,  79,  95,  100
      MOVE G6D, 90,  90, 127, 147, 116,  100
      MOVE G6B,140,  45,  70, 100, 100, 100
      MOVE G6C, 60,  45,  70, 100, 100, 100
      WAIT

      DELAY 500
      HIGHSPEED SETOFF

right_shoot3:
      SPEED 5
      MOVE G6A,115,  76, 145,  93, 102, 100
```

```
      MOVE G6D, 70,  76, 145,  93, 104, 100
      MOVE G6B,110,  45,  70, 100, 100, 100
      MOVE G6C, 90,  45,  70, 100, 100, 100
      WAIT
      RETURN

      '================================================
left_shoot:
      SPEED 4
      MOVE G6A, 70,  56, 180,  79, 102, 100
      MOVE G6D,112,  56, 180,  79, 104, 100
      MOVE G6B, 90,  45,  70, 100, 100, 100
      MOVE G6C,110,  45,  70, 100, 100, 100
      WAIT

left_shoot1:
      SPEED 6
      MOVE G6A, 90,  90, 127,  65, 116,  100
      MOVE G6D,115,  60, 180,  79,  95,  100
      MOVE G6B,140,  45,  70, 100, 100, 100
      MOVE G6C, 60,  45,  70, 100, 100, 100
      WAIT

      SPEED 15
      HIGHSPEED SETON

left_shoot2:
      MOVE G6A, 90,  90, 127, 147, 116,  100
      MOVE G6D,115,  52, 180,  79,  95,  100
      MOVE G6B, 60,  45,  70, 100, 100, 100
      MOVE G6C,140,  45,  70, 100, 100, 100
      WAIT

      DELAY 500
      HIGHSPEED SETOFF

left_shoot3:
      SPEED 5
      MOVE G6A, 70,  76, 145,  93, 104, 100
      MOVE G6D,115,  76, 145,  93, 102, 100
      MOVE G6B, 90,  45,  70, 100, 100, 100
      MOVE G6C,110,  45,  70, 100, 100, 100
      WAIT
      RETURN
```

S'inclouen els diferents subprogrames i posicions als que cal portar els servos per poder fer els atacs frontal i lateral. La velocitat i separació entre els diferents moviments ve donat pel programa de demostració tal i com s'ha dit anteriorment

```
      '================================================
      '================================================
moviment_victòria:

      SPEED 15
      MOVE G6A, 92, 100, 110, 100, 107, 100
```

```
MOVE G6D, 92, 100, 110, 100, 107, 100
MOVE G6B,190, 150,  10, 100, 100, 100
MOVE G6C,190, 150,  10, 100, 100, 100
WAIT
SPEED 15
HIGHSPEED SETON

MOVE G6B,190,  10,  75, 100, 100, 100
MOVE G6C,190, 140,  10, 100, 100, 100
WAIT
DELAY 500
MOVE G6B,190, 140,  10, 100, 100, 100
MOVE G6C,190,  10,  75, 100, 100, 100
WAIT
DELAY 500

MOVE G6A, 92, 100, 113, 100, 107, 100
MOVE G6D, 92, 100, 113, 100, 107, 100
MOVE G6B,190, 150,  10, 100, 100, 100
MOVE G6C,190, 150,  10, 100, 100, 100
WAIT

HIGHSPEED SETOFF
MOVE G6A,100, 115,  90, 110, 100, 100
MOVE G6D,100, 115,  90, 110, 100, 100
MOVE G6B,100,  80,  60, 100, 100, 100
MOVE G6C,100,  80,  60, 100, 100, 100
WAIT

GOSUB standard_pose


SPEED 9
MOVE G24, 100,  76, 145,  93, 100,  , 100, 190, 132,  ,  ,  , 190,
30,  80,  ,  ,  , 100,  76, 145,  93, 100,
WAIT

HIGHSPEED SETON
MOVE G24, 100,  76, 145,  93, 100,  , 100, 190, 132,  ,  ,  ,  89,
147,  10,  ,  ,  , 100,  76, 145,  93, 100,
WAIT
MOVE G24, 100,  76, 145,  93, 100,  , 100, 190, 132,  ,  ,  , 183,
50,  10,  ,  ,  , 100,  76, 145,  93, 100,
WAIT
HIGHSPEED SETOFF

SPEED 7
MOVE G24,  98, 167,  45,  98,  99,  , 183,  50,  10,  ,  ,  , 183,
50,  10,  ,  ,  , 102, 169,  42,  94,  99,
WAIT
MOVE G24,  98, 167,  45,  98,  99,  , 190,  50,  10,  ,  ,  , 183,
190,  96,  ,  ,  , 102, 169,  42,  94,  99,
WAIT

SPEED 9
MOVE G24,  98, 167,  45,  98,  99,  , 183, 190,  96,  ,  ,  , 190,
50,  10,  ,  ,  , 102, 169,  42,  94,  99,
WAIT
```

```
        MOVE G24, 102,  79, 133, 100,  97,  , 183, 190,  96,  ,  ,  , 190,
        50,  10,  ,  ,  , 100,  75, 136,  99, 101,
        WAIT

        RETURN
```

Per el moviment de victòria s'introdueixen les posicions a les que cal portar els servos mitjançant l'eina Catch & Play. En aquest cas s'han introduït en grups de 6 servomotor al principi i posteriorment en grups de 24. S'esperarà a que els servomotors finalitzin el moviment i es retornarà al programa principal.

B.2.9. Exercici 8 (examen)

Exercici 8 (examen). ***Es vol que el robot realitzi diverses acrobàcies per a una demostració seguint unes ordres donades per un usuari. El robot roman en posició estàndard fins a rebre una ordre mitjançant un xiulet, moment en el qual realitzarà una reverència i saludarà al públic (pujant els braços i movent les mans repetidament).Posteriorment esperarà a rebre les diferents ordres mitjançant el comandament per infraroigs.***
Si l'usuari prem el botó 1 el robot haurà de realitzar una tombarella endavant sempre i quan la distància frontal (captada per el sensor de ultrasons) sigui suficient. En cas de polsar el botó 2 es farà una roda lateral cap a la dreta, també tenint en compte que la distància lateral (captada aquesta vegada amb el sensor de infrarojos) sigui suficient. Si el robot veiés que no pot realitzar el moviment haurà de obrir els braços i tancar-los alhora que emet un so. Finalment en cas de prémer el botó 3 haurà de recolzar-se només sobre un peu i intentar volar.
En tot moment s'haurà de comprovar l'estabilitat del robot i en cas de caiguda fer-lo aixecar i repetir el mateix moviment que estava realitzant en el moment de la caiguda. L'usuari finalitzarà el programa mitjançant un altre xiulet (que el robot seguirà esperant en tot moment) fent que torni a saludar al públic i s'apagui.

```
'Declaració de constants
CONST DIST_MIN_FRONTAL = 75
CONST DIST_MIN_LAT = 65
CONST SO_MIN = 70
CONST MIN_X = 120
CONST MAX_X = 165
CONST MIN_Y = 115
CONST MAX_Y = 165
```

Es declaren les constants i els seus valors consegüents que es vol que tinguin.

```
'Declaració de variables
DIM infraroig AS INTEGER
DIM ultrasons AS INTEGER
DIM sons AS BYTE
DIM eix_x AS BYTE
DIM eix_y AS BYTE
DIM caiguda AS BYTE
DIM comandament AS BYTE
```

Es declaren les variables de tipus byte (8 bits - [0-255]) i de tipus integer (16 bits – [0-65535]) per usar-les posteriorment en el programa.

```
'Configure PTP for using with groups of servos
PTP SETON
'Configure PTP for using with all the servos in the robot
PTP ALLON
```

Activa el funcionament punt a punt de tots els servomotors.

```
'== motor direction setting ====================
DIR G6A,1,0,0,1,0,0
DIR G6B,1,1,1,1,1,1
DIR G6C,0,0,0,0,0,0
DIR G6D,0,1,1,0,1,0
```

Es defineix el sentit de gir dels diferents servomotors. Si hi ha un 1 el servomotor girarà en sentit horari mentre que un 0 farà que el servo giri en sentit antihorari.

La definició de moviments i tractaments pels servos es fa sempre per 24 motors (dels quals realment només estan 16).

Grup 6A: Cama esquerra (5 servomotors)

Grup 6B: Braç esquerre(3 servomotors)

Grup 6C: Braç dret (3 servomotors)

Grup 6D: Cama dreta (5 servomotors).

```
'== motor start position read ==================
GETMOTORSET G6A,1,1,1,1,1,0
GETMOTORSET G6B,1,1,1,0,0,0
GETMOTORSET G6C,1,1,1,0,0,0
GETMOTORSET G6D,1,1,1,1,1,0
```

Definició de la posició inicial al iniciar el programa. En cas de ser un 1 els motors es quedaran a la posició que estaven, en cas de tenir un 0 es mouran a una posició de seguretat indicada per el controladors.

```
SPEED 5
```

S'assigna a la velocitat el valor 5. Es recomana no utilitza velocitats superiors a 10 per les diferents pràctiques a realitzar

```
'== motor power on =============================
MOTOR G24
GOSUB standard_pose
```

```
'================================================
'================================================

INICI:


	'Espera fins a rebre un xiulet
	sons = AD (4)
	IF sons < SO_MIN THEN GOTO INICI

	'Realitza els moviments inicials
	GOSUB reverencia
	DELAY 1000
	GOSUB standard_pose
	DELAY 500
	GOSUB saludar
	DELAY 500
	GOSUB standard_pose
	GOTO MOVIMENTS
```

En el mode INICI, el programa està en un bucle fins que s'escolta el xiulet. En aquell moment el robot realitzarà una reverència, saludarà al públic i passarà al mode MOVIMENTS.

```
MOVIMENTS:

	'Moviments segons els diferents botons
	comandament = REMOCON (1)
	IF comandament = 1 THEN
		GOSUB mov_boto1
		DELAY 500
		GOSUB standard_pose
	ELSEIF comandament = 2 THEN
		GOSUB mov_boto2
		DELAY 500
		GOSUB standard_pose
	ELSEIF comandament = 3 THEN
		GOSUB mov_boto3
		DELAY 500
		GOSUB standard_pose
	ENDIF

	'Comprovar final
	sons = AD(4)
	IF sons >= SO_MIN THEN GOTO FINAL

	GOTO MOVIMENTS
```

En el mode MOVIMENTS es fan dues comprovacions. Primerament si l'usuari ha premut algun dels botons del controlador, fent que el robot realitzi el moviment corresponent. En segon lloc, es comprova si s'escolta un altre xiulet per tal de finalitzar el programa. El programa es quedarà en el mode MOVIMENTS indefinidament a menys que es faci un xiulet.

```
FINAL:
```

```
        'Realitzar els moviments finals
        DELAY 500
        GOSUB saludar
        DELAY 500
        GOSUB standard_pose
        DELAY 500
        GOSUB sit_pose

        END
```

Finalment quan s'ha rebut en el mode FINAL el robot saludarà al públic, s'asseurà i acabarà el programa.

```
        '================================================
        '================================================
standard_pose:
        SPEED 5
        MOVE G6A,100, 76, 145, 93, 100, 100
        MOVE G6D,100, 76, 145, 93, 100, 100
        MOVE G6B,100, 30, 80, 100, 100, 100
        MOVE G6C,100, 30, 80, 100, 100, 100
        WAIT
        RETURN
```

Es defineixen els valors de la posició estàndard (posició inicial i de seguretat).

```
    '================================================
    '================================================
reverencia:

        MOVE G6A, 100,  58, 135, 160, 100, 100
        MOVE G6D, 100,  58, 135, 160, 100, 100
        MOVE G6B, 100,  30,  80,  ,  ,  ,
        MOVE G6C, 100,  30,  80,  ,  ,  ,
        WAIT
        RETURN

    '================================================
    '================================================
saludar:

        DIM i AS BYTE

        SPEED 7
        MOVE G6A, 100,  76, 145,  93, 100,
        MOVE G6D, 100,  76, 145,  93, 100,
        MOVE G6B,  98, 185, 100,  ,  ,
        MOVE G6C,  98, 185, 100,  ,  ,
        WAIT

        DELAY 500

        SPEED 10
        FOR i = 1 TO 4
```

```
            MOVE G6A, 100,  76, 145,  93, 100,
            MOVE G6D, 100,  76, 145,  93, 100,
            MOVE G6B,  98, 185, 136,  ,  ,
            MOVE G6C,  98, 185, 136,  ,  ,
            WAIT

            MOVE G6A, 100,  76, 145,  93, 100,
            MOVE G6D, 100,  76, 145,  93, 100,
            MOVE G6B,  98, 185,  75,  ,  ,
            MOVE G6C,  98, 185,  75,  ,  ,
            WAIT

      NEXT i

      RETURN
```

Per cadascun dels subprogrames(reverència i saludar) s'introdueixen les posicions a les que cal portar els servos mitjançant l'eina Catch & Play. En aquest cas s'han introduït en grups de 6 servomotors. En el cas del subprograma per saludar es crea una variable local "i" que es farà servir per a fer les repeticions necessàries.

```
      '================================================
      '================================================
mov_boto1:

      'Mirar si la distancia es l'adequada i realitzem o no el moviment
      segons convingui
      GOSUB capturar_valor_ultrasons
      IF ultrasons > DIST_MIN_FRONTAL THEN
            GOSUB tombarella_end
            DELAY 500
            GOSUB standard_pose
      ELSEIF ultrasons <= DIST_MIN_FRONTAL THEN
            GOSUB negacio_mov
            DELAY 500
            GOSUB standard_pose
            GOSUB sit_pose
      ENDIF
```

Abans de realitzar el moviment es comprova si té suficient distància frontal mitjançant el sensor de ultrasons. En cas afirmatiu realitzarà la tombarella, sinó procedirà amb un moviment de negació i s'asseurà.

```
      'Cal comprovar si ha caigut el robot i en cas afirmatiu fer que
      torni a realitzar el moviment. Prèviament cal inicialitzar el valor
      caiguda a 0.
      caiguda = 0
      DELAY 2000
      GOSUB comprovar_caiguda
      IF caiguda = 1 THEN
            GOTO mov_boto1
      ELSEIF caiguda = 0 THEN
            RETURN
      ENDIF
```

Una vegada realitzat el moviment es comprovarà si ha caigut. Si és el cas es després d'aixecar-se (mitjançant el subprograma comprovar_caiguda) es tornarà a realitzar el moviment. En cas contrari es tornarà al programa principal. Prèviament al moviment s'inicialitza la variable caiguda a 0 i que s'activarà o no segons el valor que doni l'acceleròmetre.

```
	'================================================
	'================================================
mov_boto2:

	'Mirar si la distancia es l'adequada i realitzem o no el moviment
	segons convingui
	GOSUB capturar_valor_infraroig
	IF infraroig > DIST_MIN_LAT THEN
		GOSUB roda_dreta
		DELAY 500
		GOSUB standard_pose
	ELSEIF infraroig <= DIST_MIN_LAT THEN
		GOSUB negacio_mov
		DELAY 500
		GOSUB standard_pose
	ENDIF
```

Abans de realitzar el moviment es comprova si té suficient distància lateral mitjançant el sensor d'infraroigs. En cas afirmatiu realitzarà la roda sinó procedirà amb un moviment de negació i s'asseurà.

```
	'Cal comprovar si ha caigut el robot i en cas afirmatiu fer que
	torni a realitzar el moviment. Prèviament cal inicialitzar el valor
	caiguda a 0.
	caiguda = 0
	DELAY 2000
	GOSUB comprovar_caiguda
	IF caiguda = 1 THEN
		GOTO mov_boto2
	ELSEIF caiguda = 0 THEN
		RETURN
	ENDIF
```

Una vegada realitzat el moviment es comprovarà si ha caigut. Si és el cas es després d'aixecar-se (mitjançant el subprograma comprovar_caiguda) es tornarà a realitzar el moviment. En cas contrari es tornarà al programa principal. Prèviament al moviment s'inicialitza la variable caiguda a 0 i que s'activarà o no segons el valor que doni l'acceleròmetre.

```
	'================================================
	'================================================
mov_boto3:

	GOSUB volar_peu
	DELAY 500
	GOSUB standard_pose

	'Cal comprovar si ha caigut el robot i en cas afirmatiu fer que
	torni a realitzar el moviment. Prèviament cal inicialitzar el valor
	caiguda a 0.
```

```
        caiguda = 0
        DELAY 2000
        GOSUB comprovar_caiguda
        IF caiguda = 1 THEN
                GOTO mov_boto3
        ELSEIF caiguda = 0 THEN
                RETURN
        ENDIF
```

En aquest cas no cal comprovar cap distància i es realitza el moviment directament. Una vegada realitzat, es comprovarà si ha caigut. Si és el cas es després d'aixecar-se (mitjançant el subprograma comprovar_caiguda) es tornarà a realitzar el moviment. En cas contrari es tornarà al programa principal. Prèviament al moviment s'inicialitza la variable caiguda a 0 i que s'activarà o no segons el valor que doni l'acceleròmetre.

```
        '================================================
        '================================================
capturar_valor_infraroig:

        infraroig = AD (6)
        IF infraroig < 4 THEN infraroig = 4     'min
        infraroig = infraroig - 3
        infraroig = 6787 / infraroig

        IF infraroig < 10 OR infraroig > 150 THEN
                GOTO capturar_valor_infraroig

        WAIT
        RETURN
```

Es fa servir el subprograma explicat anteriorment per a captar el valor del sensor d'infraroigs

```
        '================================================
        '================================================
capturar_valor_ultrasons:

        ultrasons = AD(2)
        ultrasons = ultrasons * 13
        ultrasons = ultrasons / 10

        WAIT
        RETURN
```

Es capta el valor del sensor de ultrasons i es transforma a cm fent servir les comandes explicades.

```
        '================================================
        '================================================
negacio_mov:
        DIM j AS BYTE

        MUSIC "7C"
        DELAY 1000

        SPEED 7

        MOVE G6A, 100,  76, 145,  93, 100,
```

```
      MOVE G6D, 100,  76, 145,  93, 100,
      MOVE G6B, 184,  25,  59,  ,  ,
      MOVE G6C, 184,  25,  59,  ,  ,
      WAIT

      FOR j = 1 TO 3

            MOVE G6A, 100,  76, 145,  93, 100,
            MOVE G6D, 100,  76, 145,  93, 100,
            MOVE G6B, 184,  57,  59,  ,  ,
            MOVE G6C, 184,  57,  59,  ,  ,
            WAIT

            MOVE G6A, 100,  76, 145,  93, 100,
            MOVE G6D, 100,  76, 145,  93, 100,
            MOVE G6B, 184,  25,  59,  ,  ,
            MOVE G6C, 184,  25,  59,  ,  ,
        WAIT

      NEXT j

      RETURN
```

Pel moviment de negació s'introdueixen les posicions a les que cal portar els servos mitjançant l'eina Catch & Play. En aquest cas s'han introduït en grups de 6 servomotors. Es crea una variable local "j" que es farà servir per a fer les repeticions necessàries.

```
      '================================================
      '================================================
tombarella_end:

      SPEED 8
      MOVE G6A,100, 155,  20, 140, 100, 100
      MOVE G6D,100, 155,  20, 140, 100, 100
      MOVE G6B,130,  50,  85, 100, 100, 100
      MOVE G6C,130,  50,  85, 100, 100, 100
      WAIT

      MOVE G6A, 60, 165,  30, 165, 155, 100
      MOVE G6D, 60, 165,  30, 165, 155, 100
      MOVE G6B,170,  10, 100, 100, 100, 100
      MOVE G6C,170,  10, 100, 100, 100, 100
      WAIT

      MOVE G6A, 75, 165,  55, 165, 155, 100
      MOVE G6D,75, 165,  55, 165, 155, 100
      MOVE G6B,185,  10, 100, 100, 100, 100
      MOVE G6C,185,  10, 100, 100, 100, 100
      WAIT

      MOVE G6A, 80, 155,  85, 150, 150, 100
      MOVE G6D, 80, 155,  85, 150, 150, 100
      MOVE G6C,185,  40, 60,  100, 100, 100
      WAIT
```

```
MOVE G6A,100, 130, 120,  80, 110, 100
MOVE G6D,100, 130, 120,  80, 110, 100
MOVE G6B,130, 160,  10, 100, 100, 100
MOVE G6C,130, 160,  10, 100, 100, 100
WAIT

MOVE G6A,100, 160, 110, 140, 100, 100
MOVE G6D,100, 160, 110, 140, 100, 100
MOVE G6B,140,  70,  20, 100, 100, 100
MOVE G6C,140,  70,  20, 100, 100, 100
WAIT

SPEED 15
MOVE G6A,100,  56, 110,  26, 100, 100
MOVE G6D,100,  71, 177, 162, 100, 100
MOVE G6B,170,  40,  50, 100, 100, 100
MOVE G6C,170,  40,  50, 100, 100, 100
WAIT

MOVE G6A,100,  62, 110,  15, 100, 100
MOVE G6D,100,  71, 128, 113, 100, 100
MOVE G6B,190,  40,  50, 100, 100, 100
MOVE G6C,190,  40,  50, 100, 100, 100
WAIT

SPEED 15
MOVE G6A,100,  55, 110,  15, 100, 100
MOVE G6D,100,  55, 110,  15, 100, 100
MOVE G6B,190,  40,  50, 100, 100, 100
MOVE G6C,190,  40,  50, 100, 100, 100
WAIT

SPEED 10

MOVE G6A,100, 110, 100,  15, 100, 100
MOVE G6D,100, 110, 100,  15, 100, 100
MOVE G6B,170, 160, 115, 100, 100, 100
MOVE G6C,170, 160, 115, 100, 100, 100
WAIT

MOVE G6A,100, 170,  70,  15, 100, 100
MOVE G6D,100, 170,  70,  15, 100, 100
MOVE G6B,190, 170, 120, 100, 100, 100
MOVE G6C,190, 170, 120, 100, 100, 100
WAIT

MOVE G6A,100, 170,  30,  110, 100, 100
MOVE G6D,100, 170,  30,  110, 100, 100
MOVE G6B,190,  40,  60, 100, 100, 100
MOVE G6C,190,  40,  60, 100, 100, 100
WAIT

GOSUB sit_pose

GOSUB standard_pose

RETURN
'================================================
```

```
sit_pose:

      SPEED 10
      MOVE G6A,100, 151,  23, 140, 101, 100,
      MOVE G6D,100, 151,  23, 140, 101, 100
      MOVE G6B,100,  30,  80, 100, 100, 100
      MOVE G6C,100,  30,  80, 100, 100, 100
      WAIT
      RETURN

    '================================================
    '================================================
roda_dreta:


      SPEED 8
      MOVE G6A,100, 135,  60, 123, 100, 100
      MOVE G6D,100, 135,  60, 123, 100, 100
      MOVE G6B,100, 120, 140, 100, 100, 100
      MOVE G6C,100, 120, 140, 100, 100, 100
      WAIT
      DELAY 100

      SPEED 3
      MOVE G6A, 83, 110,  91, 116, 100, 100
      MOVE G6D,114, 135,  60, 123, 105, 100
      MOVE G6B,100, 120, 140, 100, 100, 100
      MOVE G6C,100, 120, 140, 100, 100, 100
      WAIT
      DELAY 100

      MOVE G6A,89,  135,  60, 123, 100, 100
      MOVE G6D,114, 135,  60, 123, 105, 100
      MOVE G6B,100, 120, 140, 100, 100, 100
      MOVE G6C,100, 120, 140, 100, 100, 100
      WAIT

      MOVE G6A, 89, 135,  60, 123, 130, 100
      MOVE G6D,120, 135,  60, 123, 110, 100
      MOVE G6B,100, 120, 140, 100, 100, 100
      MOVE G6C,100, 120, 140, 100, 100, 100
      WAIT

      SPEED 4
      MOVE G6A,89,  135,  60, 123, 158, 100
      MOVE G6D,120, 135,  60, 123, 120, 100
      MOVE G6B,100, 165, 185, 100, 100, 100
      MOVE G6C,100, 165, 185, 100, 100, 100
      WAIT

      SPEED 8
      MOVE G6A,120, 131,  60, 123, 183, 100
      MOVE G6D,120, 131,  60, 123, 185, 100
      MOVE G6B,100, 165, 185, 100, 100, 100
      MOVE G6C,100, 165, 185, 100, 100, 100
      WAIT

      DELAY 200
```

```
      SPEED 5
      MOVE G6A,120, 131,  60, 123, 183, 100
      MOVE G6D,120, 131,  60, 123, 185, 100
      MOVE G6B,100, 120, 145, 100, 100, 100
      MOVE G6C,100, 120, 145, 100, 100, 100
      WAIT

      SPEED 6
      MOVE G6A,105, 131,  60, 123, 183, 100
      MOVE G6D, 86, 112,  73, 127, 101, 100
      MOVE G6B,100, 120, 145, 100, 100, 100
      MOVE G6C,100, 120, 145, 100, 100, 100
      WAIT

      SPEED 3
      MOVE G6A,112, 131,  62, 123, 133, 100
      MOVE G6D, 86, 118,  73, 127, 101, 100
      MOVE G6B,100, 80, 80, 100, 100, 100
      MOVE G6C,100, 80, 80, 100, 100, 100
      WAIT

      SPEED 3
      MOVE G6A,107, 135,  62, 123, 113, 100
      MOVE G6D, 88, 115,  89, 115,  90, 100
      MOVE G6B,100, 80, 80, 100, 100, 100
      MOVE G6C,100, 80, 80, 100, 100, 100
      WAIT

      SPEED 4
      MOVE G6A,100, 135,  60, 123, 100, 100
      MOVE G6D,100, 135,  60, 123, 100, 100
      MOVE G6B,100, 80, 80, 100, 100, 100
      MOVE G6C,100, 80, 80, 100, 100, 100
      WAIT

      RETURN

    '================================================
    '================================================
volar_peu:

      DIM k AS BYTE
      SPEED 5

      MOVE G6A, 85,  71, 152,  91, 112, 60
      MOVE G6D,112,  76, 145,  93,  92, 60
      MOVE G6B,100,  40,  80, , , ,

      MOVE G6C,100,  40,  80, , , ,
      WAIT

      MOVE G6A, 90,  98, 105,  115, 115, 60
      MOVE G6D,116,  74, 145,  98,  93, 60
      MOVE G6B,100,  150,  150, 100, 100, 100
      MOVE G6C,100,  150,  150, 100, 100, 100
      WAIT
```

```
MOVE G6A, 90, 121,  36, 105, 115,  60
MOVE G6D,116,  60, 146, 138,  93,  60
MOVE G6B,100,  150,  150, 100, 100, 100
MOVE G6C,100,  150,  150, 100, 100, 100
WAIT

MOVE G6A, 90,  98, 105,  64, 115,  60
MOVE G6D,116,  50, 160, 160,  93,  60
MOVE G6B,145, 110, 110, 100, 100, 100
MOVE G6C,145, 110, 110, 100, 100, 100
WAIT

FOR k = 10 TO 15
      SPEED i
      MOVE G6B,145,  80,  80, 100, 100, 100
      MOVE G6C,145,  80,  80, 100, 100, 100
      WAIT

      MOVE G6B,145,  120,  120, 100, 100, 100
      MOVE G6C,145,  120,  120, 100, 100, 100
      WAIT
NEXT k

DELAY 1000
SPEED 6

MOVE G6A, 90,  98, 105,  64, 115,  60
MOVE G6D,116,  50, 160, 160,  93,  60
MOVE G6B,100, 160, 180, 100, 100, 100
MOVE G6C,100, 160, 180, 100, 100, 100
WAIT


MOVE G6A, 90, 121,  36, 105, 115,  60
MOVE G6D,116,  60, 146, 138,  93,  60
MOVE G6B,100,  150,  150, 100, 100, 100
MOVE G6C,100,  150,  150, 100, 100, 100
WAIT
SPEED 4

MOVE G6A, 90,  98, 105,  115, 115, 60
MOVE G6D,116,  74, 145,  98,  93, 60
WAIT

MOVE G6A, 85,  71, 152,  91, 112, 60
MOVE G6D,112,  76, 145,  93,  92, 60
MOVE G6B,100,  40,  80, , , ,
MOVE G6C,100,  40,  80, , , ,
WAIT

RETURN
```

S'inclouen els diferents subprogrames i posicions als que cal portar els servos tant per la tombarella endavant com per la roda i el moviment de volar amb un peu. En últim cas es fa ús també d'una variable local "k" per fer les diferents repeticions. La velocitat i separació entre els diferents moviments ve donat pel programa de demostració tal i com s'ha dit anteriorment.

```
	'================================================
	'================================================
comprovar_caiguda:

	eix_x = AD(1)
	IF eix_x <= MIN_X THEN
		GOSUB aixecar_esquerra
		caiguda = 1
	ELSEIF eix_x >= MAX_X THEN
		GOSUB aixecar_dreta
		caiguda = 1
	ENDIF


	eix_y = AD(0)
	IF eix_y <= MIN_Y THEN
		GOSUB aixecar_enrere
		caiguda = 1
	ELSEIF eix_y >= MAX_Y THEN
		GOSUB aixecar_endav
		caiguda = 1
	ENDIF

	GOSUB standard_pose

	RETURN
```

Es comprova la caiguda primer a l'eix x i després a l'eix y mitjançant el valor captat per l'acceleròmetre. En cas de caiguda s'executarà el subprograma per aixecar que correspongui i s'activarà la variable de caiguda. Finalment es comprova l'estat de la variable caiguda, en cas de ser 0 es tornarà al programa principal mentre que si està activa es passarà al subprograma que apagarà el robot.

```
	'================================================
	'================================================
aixecar_dreta:
	SPEED 10

	MOVE G6A, 100,  76, 145, 180, 100,
	MOVE G6D, 100,  76, 145,  57, 100,
	MOVE G6B, 100,  30,  80,  ,  ,
	MOVE G6C, 100,  30,  80,  ,  ,

	MOVE G6A,100, 76, 145, 93, 100, 100
	MOVE G6D,100, 76, 145, 93, 100, 100
	MOVE G6B,100, 30, 80, 100, 100, 100
	MOVE G6C,100, 30, 80, 100, 100, 100

	WAIT
	RETURN

aixecar_esquerra:
	SPEED 10
	MOVE G6A, 100,  76, 145,  93, 100,
	MOVE G6D, 100,  76, 145,  14, 100,
	MOVE G6B, 100,  30,  80,  ,  ,
	MOVE G6C, 100,  30,  80,  ,  ,
```

```
      MOVE G6A,100, 76, 145, 93, 100, 100
      MOVE G6D,100, 76, 145, 93, 100, 100
      MOVE G6B,100, 30, 80, 100, 100, 100
      MOVE G6C,100, 30, 80, 100, 100, 100

      WAIT
      RETURN
```

En cas de caiguda lateral, degut a que els braços no tenen suficient força per aixecar el robot, el que es fa es moure el braç contrari endavant o enrere de forma que el robot es desequilibri i passi d'una posició de caiguda lateral a caiguda frontal o posterior, respectivament.

```
      '================================================
      '================================================
aixecar_endav:

      SPEED 10

      MOVE G6A,100, 130, 120,  80, 110, 100
      MOVE G6D,100, 130, 120,  80, 110, 100
      MOVE G6B,150, 160,  10, 100, 100, 100
      MOVE G6C,150, 160,  10, 100, 100, 100
      WAIT

      MOVE G6A, 80, 155,  85, 150, 150, 100
      MOVE G6D, 80, 155,  85, 150, 150, 100
      MOVE G6B,185,  40, 60,  100, 100, 100
      MOVE G6C,185,  40, 60,  100, 100, 100
      WAIT

      MOVE G6A, 75, 165,  55, 165, 155, 100
      MOVE G6D, 75, 165,  55, 165, 155, 100
      MOVE G6B,185,  10, 100, 100, 100, 100
      MOVE G6C,185,  10, 100, 100, 100, 100
      WAIT

      MOVE G6A, 60, 165,  30, 165, 155, 100
      MOVE G6B,170,  10, 100, 100, 100, 100
      MOVE G6C,170,  10, 100, 100, 100, 100
      WAIT

      MOVE G6A, 60, 165,  25, 160, 145, 100
      MOVE G6D, 60, 165,  25, 160, 145, 100
      MOVE G6B,150,  60,  90, 100, 100, 100
      MOVE G6C,150,  60,  90, 100, 100, 100
      WAIT

      MOVE G6A,100, 155,  25, 140, 100, 100
      MOVE G6D,100, 155,  25, 140, 100, 100
      MOVE G6B,130,  50,  85, 100, 100, 100
      MOVE G6C,130,  50,  85, 100, 100, 100
      WAIT

      RETURN
```

```
aixecar_enrere:

      SPEED 10

      MOVE G6A,100,  10, 100, 115, 100, 100
      MOVE G6D,100,  10, 100, 115, 100, 100
      MOVE G6B,100, 130,  10, 100, 100, 100
      MOVE G6C,100, 130,  10, 100, 100, 100
      WAIT

      MOVE G6A,100, 10,  83, 140, 100, 100
      MOVE G6D,100, 10,  83, 140, 100, 100
      MOVE G6B,20, 130,  10, 100, 100, 100
      MOVE G6C,20, 130,  10, 100, 100, 100
      WAIT

      MOVE G6A,100, 126,  60, 50, 100, 100
      MOVE G6D,100, 126,  60, 50, 100, 100
      MOVE G6B,20,  30,  90, 100, 100, 100
      MOVE G6C,20,  30,  90, 100, 100, 100
      WAIT

      MOVE G6A,100, 165,  70, 15, 100, 100
      MOVE G6D,100, 165,  70, 15, 100, 100
      MOVE G6B, 30,  20,  95,100, 100, 100
      MOVE G6C, 30,  20,  95,100, 100, 100
      WAIT

      MOVE G6A,100, 165,  40, 100, 100, 100
      MOVE G6D,100, 165,  40, 100, 100, 100
      MOVE G6B,110,  70,  50, 100, 100, 100
      MOVE G6C,110,  70,  50, 100, 100, 100
      WAIT

      RETURN
```

S'inclouen els diferents subprogrames i posicions als que cal portar els servos per poder aixecar-se endavant o enrere. La velocitat i separació entre els diferents moviments ve donat pel programa de demostració tal i com s'ha dit anteriorment.

B.3. Exercicis de Visual Servoing

En aquest apartat s'inclouran tant el codi de Matlab del GUI de la pràctica 4 de Visual Servoing com també la solució de l'exercici en Robonova que els alumnes hauran de dissenyar per tal de fer funcionar el muntatge comentat a la pràctica.

Per tal de realitzar poder fer ús dels dispositius en YS-C10U per a la comunicació mitjançant radiofreqüència, s'ha de fer un circuit d'adaptació (MAX 232) del sèrie del dispositiu al sèrie

del robot. L'esquema és el mateix que el corresponent al cable utilitzat per al calibratge dels sensors i que es pot veure a la figura B.1.

B.3.6. Codi Matlab

```
function varargout = GUI_final_prova(varargin)

% GUI_FINAL_PROVA MATLAB code for GUI_final_prova.fig

%      GUI_FINAL_PROVA, by itself, creates a new GUI_FINAL_PROVA or

       raises the existing singleton*.

%

%      H = GUI_FINAL_PROVA returns the handle to a new GUI_FINAL_PROVA or

       the handle to the existing singleton*.

%

%      GUI_FINAL_PROVA('CALLBACK',hObject,eventData,handles,...) calls

       the local function named CALLBACK in GUI_FINAL_PROVA.M with the

       given input arguments.

%

%      GUI_FINAL_PROVA('Property','Value',...) creates a new

       GUI_FINAL_PROVA or raises the existing singleton*.  Starting from

       the left, property value pairs are applied to the GUI before

       GUI_final_prova_OpeningFcn gets called. An unrecognized property

       name or invalid value makes property application stop.  All inputs

       are passed to GUI_final_prova_OpeningFcn via varargin.

%

%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only

       One instance to run (singleton)".
```

```
%
% See also: GUIDE, GUIDATA, GUIHANDLES


% Edit the above text to modify the response to help GUI_final_prova


% Last Modified by GUIDE v2.5 14-Oct-2011 13:17:54


% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @GUI_final_prova_OpeningFcn, ...
                   'gui_OutputFcn',  @GUI_final_prova_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end


if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
```

```
% End initialization code - DO NOT EDIT


% --- Executes just before GUI_final_prova is made visible.
function GUI_final_prova_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to GUI_final_prova (see VARARGIN)



%Inicialització dels diferents axes i botons del programa
cla(handles.Real_time,'reset');
axis (handles.Real_time,'off');


handles.path = 1;
background = imread('Recta.bmp');
imshow(background,'Parent',handles.Preview_path);
axis(handles.Preview_path,'off');


set(handles.boto_path,'String','PATH ON');
set(handles.boto_trace,'String','TRACE ON');
```

```
%Inicialització de les variables que es fan servir durant el programa
handles.vidobj = videoinput('winvideo',1,'RGB24_640x480');
set(handles.boto_start,'Value',0);
handles.canalcom=serial('COM5');
set(handles.canalcom,'Timeout',20);
handles.comptador = 1;
handles.final_trajecte=0;



% Choose default command line output for GUI_final_prova
handles.output = hObject;


% Update handles structure
guidata(hObject, handles);


% UIWAIT makes GUI_final_prova wait for user response (see UIRESUME)
% uiwait(handles.figure1);



% --- Outputs from this function are returned to the command line.
function varargout = GUI_final_prova_OutputFcn(hObject, eventdata,
handles)
% varargout  cell array for returning output args (see VARARGOUT);
```

```
% hObject    handle to figure

% eventdata  reserved - to be defined in a future version of MATLAB

% handles    structure with handles and user data (see GUIDATA)



% Get default command line output from handles structure

varargout{1} = handles.output;




function Coordenades_Callback(hObject, eventdata, handles)

% hObject    handle to Coordenades (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB

% handles    structure with handles and user data (see GUIDATA)



% Hints: get(hObject,'String') returns contents of Coordenades as text

%        str2double(get(hObject,'String')) returns contents of
Coordenades as a double




% --- Executes during object creation, after setting all properties.

function Coordenades_CreateFcn(hObject, eventdata, handles)

% hObject    handle to Coordenades (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB

% handles    empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on Windows.

%       See ISPC and COMPUTER.

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end




function Instruccions_Callback(hObject, eventdata, handles)

% hObject    handle to Instruccions (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB

% handles    structure with handles and user data (see GUIDATA)



% Hints: get(hObject,'String') returns contents of Instruccions as text

%        str2double(get(hObject,'String')) returns contents of
Instruccions as a double




% --- Executes during object creation, after setting all properties.

function Instruccions_CreateFcn(hObject, eventdata, handles)

% hObject    handle to Instruccions (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB

% handles    empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on Windows.

%       See ISPC and COMPUTER.

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end




 function Errors_robot_Callback(hObject, eventdata, handles)

% hObject    handle to Errors_robot (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB

% handles    structure with handles and user data (see GUIDATA)



% Hints: get(hObject,'String') returns contents of Errors_robot as text

%        str2double(get(hObject,'String')) returns contents of
Errors_robot as a double




 % --- Executes during object creation, after setting all properties.

function Errors_robot_CreateFcn(hObject, eventdata, handles)

% hObject    handle to Errors_robot (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB

% handles    empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on Windows.

%       See ISPC and COMPUTER.

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

    set(hObject,'BackgroundColor','white');

end




% --- Executes on button press in boto_path.

function boto_path_Callback(hObject, eventdata, handles)

% hObject    handle to boto_path (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB

% handles    structure with handles and user data (see GUIDATA)


% Hint: get(hObject,'Value') returns toggle state of boto_path


if (get(handles.boto_path,'Value') == 1)

      set(handles.boto_path,'String','PATH ON');

elseif (get(handles.boto_path,'Value') == 0)

      set(handles.boto_path,'String','PATH OFF');

end

guidata(hObject,handles);




 % --- Executes on button press in boto_trace.
```

```
function boto_trace_Callback(hObject, eventdata, handles)
% hObject    handle to boto_trace (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of boto_trace

if (get(handles.boto_trace,'Value') == 1)
      set(handles.boto_trace,'String','TRACE ON');
elseif (get(handles.boto_trace,'Value') == 0)
      set(handles.boto_trace,'String','TRACE OFF');
end
guidata(hObject,handles);



% --- Executes on button press in boto_start.
function boto_start_Callback(hObject, eventdata, handles)
% hObject    handle to boto_start (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%Canviar l'estat del programa + Obrir el port de comunicacions
set(handles.boto_start,'Value',1);
fopen(handles.canalcom);
```

```
start(handles.vidobj);

OK=98;

RESET=99;

handles.comptador = 1;

handles.final_trajecte=0;

guidata(hObject,handles)


%Inicialització dels axes

cla(handles.Real_time,'reset');

axis (handles.Real_time,'off');

cla(handles.Preview_path,'reset');

axis(handles.Preview_path,'off');


%Guardar imatges de la trajectoria i representacio si es vol

if handles.path == 1

      T = imread('Recta.bmp');

      J = imread('Recta_colors.bmp');

elseif handles.path == 2

      T = imread('Corva.bmp');

      J = imread('Corva_colors.bmp');

elseif handles.path == 3

      T = imread('Oval.bmp');

      J = imread('Oval_colors.bmp');

elseif handles.path == 4
```

```
      T = imread('Forma_V.bmp');

      J = imread('Forma_V_colors.bmp');

elseif handles.path == 5

      T = imread('Forma_D.bmp');

      J = imread('Forma_D_colors.bmp');

elseif handles.path == 6

      T = imread('Forma_S.bmp');

      J = imread('Forma_S_colors.bmp');

elseif handles.path == 7

      T = imread('Forma_Anec.bmp');

      J = imread('Forma_Anec_colors.bmp');

end



%Representacio del camí si es vol

OPT=imshow(T,'Parent',handles.Real_time);

axis (handles.Real_time,'off');

if (get(handles.boto_path,'Value') == 1)

            set(OPT,'Visible','on');

elseif (get(handles.boto_path,'Value') == 0)

      set(OPT,'Visible','off');

end

hold (handles.Real_time,'on');



%Extraccio carecteristiques de la trajectoria
```

```
level_path=graythresh(T);

P=im2bw(T,level_path);

P=~P;

DP=bwdist(P);



%Crear mapa de colors per les diferents direccions sobre la trajectoria

% 0-negre(N) 1-blau(NE) 2-cian(E) 3-vermell(SE) 4-gris(S) % 5-verd(SW) 6-
magenta(W) 7-groc(NW) 8-blanc(out of path) 9-verd militar (end of path)

map = [0 0 0; 0 0 1; 0 1 1; 1 0 0; 0.894 0.894 0.894; 0 1 0; 1 0 1; 1 1
0; 1 1 1; 0.502 0.502 0];

Z=rgb2ind(J,map);



%Bucle del programa

while (get(handles.boto_start,'Value') == 1)



      %Capturar una imatge del video

      I = getsnapshot(handles.vidobj);



      %TRACTAMENT DE LA IMATGE

      %1-Binaritzacio

      level=graythresh(I);

      BW=im2bw(I,level);

      %2-Tractament morfologic

      SE = strel('square',3);

      BW2=imerode(BW,SE);
```

```
BWf=imdilate(BW2,SE);
%3-Etiquetat i canvi del background
L=bwlabel(BWf);
for i=1:480
      for j=1:640
            if L(i,j)== 0
                L(i,j) = 3;
            end
            if L(i,j)== 1
                L(i,j) = 0;
            end
            if L(i,j)== 3
                L(i,j) = 1;
            end
      end
end
%EXTRACCIO DE CARECTERISTIQUES
%Trobar el punt de direcció (m,n)
ctd = regionprops(L, 'centroid');
centroids = cat(1, ctd.Centroid);
m=centroids(2,2);
n=centroids(2,1);
%Trobar el centre del robot (x,y)
x=centroids(1,2)
y=centroids(1,1)
```

```
%Localitzar el centre del robot respecte la trajectoria i realitzar
accions segons pertoqui.
error_posició = DP(round(x),round(y));
%Obtenir vector direcció robot
v=[m-x,n-y];
[theta,~]=cart2pol(v(1),v(2));
theta=radtodeg(theta)
if theta < 0
      theta=theta+360
end

%Actualitzar la representacio del camí i el recorregut del robot
segons pertoqui
if handles.comptador == 1
      x_trace=round(x);
      y_trace=round(y);
elseif handles.comptador > 1
      x_trace=cat(1,x_trace,round(x));
      y_trace=cat(1,y_trace,round(y));
end
OTR=plot(handles.Real_time,y_trace,x_trace,'r-',y_trace,x_trace,
'rs','MarkerSize',7,'LineWidth',1.25);
if (get(handles.boto_trace,'Value') == 1)
      set(OTR,'Visible','on');
```

```
elseif (get(handles.boto_trace,'Value') == 0)
      set(OTR,'Visible','off');
end
if (get(handles.boto_path,'Value') == 1)
      set(OPT,'Visible','on');
elseif (get(handles.boto_path,'Value') == 0)
      set(OPT,'Visible','off');
end
hold (handles.Real_time,'on');



%Cas a - El robot està sobre el cami
if error_posició == 0
      %ACCIO DE CONTROL
      %Obtenir direcció a seguir pel robot segons el mapa de colors
      creat, si estem al final aturar programa.
      % 0-negre(N) 1-blau(NE) 2-cian(E) 3-vermell(SE) 4-gris(S)
      5- verd(SW) %6-magenta(W) 7-groc(NW) 9-blanc(end of path)
      if Z(round(x),round(y)) == 0
            beta=180
      elseif Z(round(x),round(y)) == 1
            beta=135
      elseif Z(round(x),round(y)) == 2
            beta=90
```

```
        elseif Z(round(x),round(y)) == 3
            beta=45
        elseif Z(round(x),round(y)) == 4
            beta=0
        elseif Z(round(x),round(y)) == 5
            beta=315
        elseif Z(round(x),round(y)) == 6
            beta=270
        elseif Z(round(x),round(y)) == 7
            beta=225
        elseif Z(round(x),round(y)) == 9
            % Activar final de trajecte + Canviar l'estat del
            programa + Tancar comunicacions
            handles.final_trajecte = 1;
            set(handles.boto_start,'Value',0);
            fclose(handles.canalcom);
            stop(handles.vidobj);
            %Borrat de variables
            clear OK RESET final_trajecte J;
            clear T OP level_path P DP map;
            clear Z I level BW SE BW2;
            clear BWf L i j ctd centroids;
            clear m n x y error_posició theta;
            clear x_trace y_trace beta error_orientacio ordre_gir
            quant_gir;
```

```
            clear ordre_cam quant_cam coord errors instruc C;
            clear D INTER min a b w
            clear alpha background;;
      end
      if handles.final_trajecte == 0
            %Trobar l'error a l'orientació i tractar-lo
            error_orientacio=beta-theta;
            if abs(error_orientacio) > 180
                  if error_orientacio > 0
                        error_orientacio = error_orientacio - 360;
                  elseif error_orientacio < 0
                        error_orientacio = error_orientacio + 360;
                  end
            end

            %Introduccio dels errors al controlador i obtenir les
            ordres a enviar.
            if error_orientacio < 0
                  ordre_gir = 2;
                  quant_gir = abs(round(error_orientacio/22.81));
                  %Cada gir a la dreta es de 22.81 de mitjaº
            elseif error_orientacio > 0
                  ordre_gir = 3;
                  quant_gir = (round(error_orientacio/24.37));
```

```
            %Cada gir a l'esquerra es de 24.37º
        end
        %Al estar al camí només fem un pas
        ordre_cam=1;
        quant_cam=1;
        %ESCRIURE VALORS A LES DIFERENTS FINESTRES
        (Coordenades,errors posició i orientacio, instruccions)
        if handles.comptador == 1
            coord=sprintf('X:%d Y:%d',round(x),round(y));
            errors=sprintf('Error posició: %d \nError
            orientacio:%d \n',round(error_posició),
            round(error_orientacio));
            instruc=sprintf('Ordre gir: %d \nNumero de girs:
            %d nOrdre caminar: %d \nNumero de passos: %d \n',
            ordre_gir, quant_gir,ordre_cam,quant_cam);
        elseif handles.comptador > 1
            coord=char(coord,sprintf('X: %d  Y: %d' ,
            round(x),round(y)));

            errors=char(errors,sprintf('Error posició: %d
            \nError orientacio: %d \n', round(error_posició),
            round(error_orientacio)));
            instruc=char(instruc,sprintf('Ordre gir: %d
            \nNumero de girs: %d \nOrdre caminar: %d \nNumero
            de passos: %d \n', ordre_gir, quant_gir,
```

```
            ordre_cam,quant_cam));
        end
        set(handles.Coordenades,'String',coord);
        assignin('base','Coordenades',coord);
        set(handles.Errors_robot,'String',errors);
        assignin('base','Errors',errors);
        set(handles.Instruccions,'String',instruc);
        assignin('base','Instruccions',instruc);
        %ENVIAR LES ORDRES AL ROBOT (si hi ha error s'envia de
         nou)
        %Girar
        if quant_gir > 0
            valor_reb=0;
            fwrite(handles.canalcom,ordre_gir,'uint8');
            valor_reb=fread(handles.canalcom,1,'uint8');
            while valor_reb ~= OK
                fwrite(handles.canalcom,ordre_gir,'uint8');
                valor_reb=fread(handles.canalcom,1,
                'uint8');
            end
            valor_reb=0;
            fwrite(handles.canalcom,quant_gir,'uint8');
            valor_reb=fread(handles.canalcom,1,'uint8');
            while valor_reb ~= OK
                fwrite(handles.canalcom,quant_gir,'uint8');
```

```
                        valor_reb=fread(handles.canalcom,1,
                        'uint8');
                end
                %Rebre valor de reset
                valor_reb = 0;
                valor_reb=fread(handles.canalcom,1,'uint8');
                while valor_reb ~= RESET
                        valor_reb=fread(handles.canalcom,1,
                        'uint8');
                end
        end
        %Caminar
        valor_reb=0;
        fwrite(handles.canalcom,ordre_cam,'uint8');
        valor_reb=fread(handles.canalcom,1,'uint8');
        while valor_reb ~= OK
                fwrite(handles.canalcom,ordre_cam,'uint8');
                valor_reb=fread(handles.canalcom,1,'uint8');
        end
        valor_reb=0;
        fwrite(handles.canalcom,quant_cam,'uint8');
        valor_reb=fread(handles.canalcom,1,'uint8');
        while valor_reb ~= OK
                fwrite(handles.canalcom,quant_cam,'uint8');
                valor_reb=fread(handles.canalcom,1,'uint8');
```

```
            end
            %Rebre valor de reset
            valor_reb = 0;
            valor_reb=fread(handles.canalcom,1,'uint8');
            while valor_reb ~= RESET
                  valor_reb=fread(handles.canalcom,1,'uint8');
            end
      end


      %Cas b - El robot està fora del camí
      elseif error_posició > 0
      %ACCIO DE CONTROL
      %Realitzem la matriu del centroide, la matriu de distàncies i
      fem la interseccio per trobar el punt més proper (a,b)
      C=zeros(480,640);
      C(round(x),round(y))=1;
      D=bwdist(C);
      INTER=P.*D;
      min = 800; %Max distancia en una imatge 480x640
      for i=1:480
            for j=1:640
                  if (INTER(i,j) > 0) && INTER(i,j) < min
                        min=INTER(i,j);
                        a=i;
                        b=j;
```

```
                end
            end
        end
        %Obtenir vector direcció a seguir per a tornar al camí
        w=[a-x,b-y];
        [alpha,~]=cart2pol(w(1),w(2));
        alpha=radtodeg(alpha)
        if alpha < 0
            alpha=alpha+360
        end
        %Trobar l'error d'orientacio i tractar-lo
        error_orientacio=round(alpha)-round(theta);
        if abs(error_orientacio) > 180
            if error_orientacio > 0
                error_orientacio = error_orientacio - 360;
            elseif error_orientacio < 0
                error_orientacio = error_orientacio + 360;
            end
        end
        %Introduccio dels errors al controlador i obtenir les ordres
        a enviar.
        if error_orientacio < 0
            ordre_gir = 2;
            quant_gir = abs(round(error_orientacio/22.81)); %Cada
            gir a la dreta es de 22.81 de mitjaº
```

```
elseif error_orientacio >= 0
      ordre_gir = 3;
      quant_gir = (round(error_orientacio/24.37)); %Cada gir
      a l'esquerra es de 24.37º
end
if error_posició < 16 % Si estem fora del cami pero propers
(menys de mig pas), realitzarem un pas complet per tornar al
camí
      quant_cam = 1;
elseif error_posició >= 16
      quant_cam = round(error_posició/30.42); %Cada pas es de
      30.42 pixels de mitja
end
ordre_cam=1;
%ESCRIURE VALORS A LES DIFERENTS FINESTRES (Coordenades,
errors posició i orientacio, instruccions)
if handles.comptador == 1
      coord=sprintf('X: %d Y: %d',round(x),round(y));

      errors=sprintf('Error posició: %d \nError orientacio:
      %d \n', round(error_posició), round(error_orientacio));
      instruc=sprintf('Ordre gir: %d \nNumero de girs: %d
      \nOrdre caminar: %d \nNumero de passos: %d \n',
      ordre_gir, quant_gir,ordre_cam,quant_cam);
elseif handles.comptador > 1
```

```
                coord=char(coord,sprintf('X: %d  Y: %d',round(x),
                round(y)));
                errors=char(errors,sprintf('Error posició: %d \nError
                orientacio: %d \n', round(error_posició), round
                (error_orientacio)));
                instruc=char(instruc,sprintf('Ordre gir: %d \nNumero de
                girs: %d \nOrdre caminar: %d \nNumero de passos: %d\n',
                ordre_gir, quant_gir,ordre_cam,quant_cam));
        end
        set(handles.Coordenades,'String',coord);
        assignin('base','Coordenades',coord);
        set(handles.Errors_robot,'String',errors);
        assignin('base','Errors',errors);
        set(handles.Instruccions,'String',instruc);
        assignin('base','Instruccions',instruc);
        %ENVIAR LES ORDRES AL ROBOT (si hi ha error s'envia de nou)
        %Girs
        if quant_gir > 0
                valor_reb=0;
                fwrite(handles.canalcom,ordre_gir,'uint8');
                valor_reb=fread(handles.canalcom,1,'uint8');
                while valor_reb ~= OK
                        fwrite(handles.canalcom,ordre_gir,'uint8');
                        valor_reb=fread(handles.canalcom,1,'uint8');
                end
```

```
                valor_reb=0;
                fwrite(handles.canalcom,quant_gir,'uint8');
                valor_reb=fread(handles.canalcom,1,'uint8');
                while valor_reb ~= OK
                        fwrite(handles.canalcom,quant_gir,'uint8');
                        valor_reb=fread(handles.canalcom,1,'uint8');
                end
                %Rebre valor de reset
                valor_reb=0;
                valor_reb=fread(handles.canalcom,1,'uint8');
                while valor_reb ~= RESET
                        valor_reb=fread(handles.canalcom,1,'uint8');
                end
        end
        %Caminar
        valor_reb=0;
        fwrite(handles.canalcom,ordre_cam,'uint8');
        valor_reb=fread(handles.canalcom,1,'uint8');
        while valor_reb ~= OK
                fwrite(handles.canalcom,ordre_cam,'uint8');
                valor_reb=fread(handles.canalcom,1,'uint8');
        end
        valor_reb=0;
        fwrite(handles.canalcom,quant_cam,'uint8');
        valor_reb=fread(handles.canalcom,1,'uint8');
```

```
            while valor_reb ~= OK
                  fwrite(handles.canalcom,quant_cam,'uint8');
                  valor_reb=fread(handles.canalcom,1,'uint8');
            end
            %Rebre valor de reset
            valor_reb = 0;
            valor_reb=fread(handles.canalcom,1,'uint8');
            while valor_reb ~= RESET
                  valor_reb=fread(handles.canalcom,1,'uint8');
            end
      end
      handles.comptador = handles.comptador + 1
      guidata(hObject,handles);
end



% --- Executes on button press in boto_stop.
function boto_stop_Callback(hObject, eventdata, handles)
% hObject    handle to boto_stop (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% + Tancar comunicacions + Canviar l'estat del programa
if (get(handles.boto_start,'Value') == 1 )
```

```
        set(handles.boto_start,'Value',0);

        fclose(handles.canalcom);

        stop(handles.vidobj);

end

%Borrat de variables

clear OK RESET final_trajecte J;

clear T OP level_path P DP map;

clear Z I level BW SE BW2;

clear BWf L i j ctd centroids;

clear m n x y error_posició theta;

clear x_trace y_trace beta error_orientacio ordre_gir quant_gir;

clear ordre_cam quant_cam coord errors instruc C;

clear D INTER min a b w;

clear alpha background;


guidata(hObject,handles);




% --- Executes on button press in boto_record.

function boto_record_Callback(hObject, eventdata, handles)

% hObject    handle to boto_record (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB

% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of boto_trace
```

```
% --- Executes when selected object is changed in seleccio_trajectories.
function seleccio_trajectories_SelectionChangeFcn(hObject, eventdata,
handles)
% hObject    handle to the selected object in seleccio_trajectories
% eventdata  structure with the following fields (see UIBUTTONGROUP)
%   EventName: string 'SelectionChanged' (read only)
%   OldValue: handle of the previously selected object or empty if none
was selected
%   NewValue: handle of the currently selected object
% handles    structure with handles and user data (see GUIDATA)


%Seleccionem la trajectoria només quan el programa esta aturat
if(get(handles.boto_start,'Value') == 0 )
      if (hObject==handles.path1)
            handles.path = 1;
            background = imread('Recta.bmp');
      elseif(hObject==handles.path2)
            handles.path = 2;
            background = imread('Corva.bmp');
      elseif(hObject==handles.path3)
            handles.path = 3;
            background = imread('Oval.bmp');
```

```
        elseif(hObject==handles.path4)
                handles.path = 4;
                background = imread('Forma_V.bmp');
        elseif(hObject==handles.path5)
                handles.path = 5;
                background = imread('Forma_D.bmp');
        elseif(hObject==handles.path6)
                handles.path = 6;
                background = imread('Forma_S.bmp');
        elseif(hObject==handles.path7)
                handles.path = 7;
                background = imread('Forma_Anec.bmp');
        end
        imshow(background,'Parent',handles.Preview_path);
        axis (handles.Preview_path,'off');
end


guidata(hObject, handles);




% --- Executes during object creation, after setting all properties.
function seleccio_trajectories_CreateFcn(hObject, eventdata, handles)
% hObject    handle to seleccio_trajectories (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    empty - handles not created until after all CreateFcns
called




% --- Executes during object creation, after setting all properties.

function Preview_path_CreateFcn(hObject, eventdata, handles)

% hObject    handle to Preview_path (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB

% handles    empty - handles not created until after all CreateFcns
called



% Hint: place code in OpeningFcn to populate Preview_path
```

B.3.7. Codi Robonova

Aquest és el codi que els estudiants hauran de realitzar dins la pràctica després d'haver llegit i entès el codi de Matlab.

```
'Declaració de constants
CONST RESET = 99
CONST OK = 98
CONST LIMIT_MOV = 12
```

Es declaren les constants i els seus valors consegüents que es vol que tinguin.

```
'Declaració de variables
DIM valor_reb AS BYTE
DIM ordre AS BYTE
DIM passos AS BYTE
DIM girs AS BYTE
DIM quantitat_mov AS BYTE
```

Es declaren les variables de tipus byte (8 bits - [0-255]) i de tipus integer (16 bits – [0-65535]) per usar-les posteriorment en el programa.

```
'Configure PTP for using with groups of servos
PTP SETON
'Configure PTP for using with all the servos in the robot
```

```
PTP ALLON
```

Activa el funcionament punt a punt de tots els servomotors.

```
'== motor diretion setting =====================
DIR G6A,1,0,0,1,0,0
DIR G6B,1,1,1,1,1,1
DIR G6C,0,0,0,0,0,0
DIR G6D,0,1,1,0,1,0
```

Es defineix el sentit de gir dels diferents servomotors. Si hi ha un 1 el servomotor girarà en sentit horari mentre que un 0 farà que el servo giri en sentit antihorari.

La definició de moviments i tractaments pels servos es fa sempre per 24 motors (dels quals realment només estan 16).

Grup 6A: Cama esquerra (5 servomotors)

Grup 6B: Braç esquerre(3 servomotors)

Grup 6C: Braç dret (3 servomotors)

Grup 6D: Cama dreta (5 servomotors)

```
'== motor start position read ==================
GETMOTORSET G6A,1,1,1,1,1,0
GETMOTORSET G6B,1,1,1,0,0,0
GETMOTORSET G6C,1,1,1,0,0,0
GETMOTORSET G6D,1,1,1,1,1,0
```

Definició de la posició inicial al iniciar el programa. En cas de ser un 1 els motors es quedaran a la posició que estaven, en cas de tenir un 0 es mouran a una posició de seguretat indicada per el controladors.

```
ZERO G6A,100,  93,  97, 107,  86, 100
ZERO G6B,105, 102, 101, 100, 100, 100
ZERO G6C, 99,  99,  99, 100, 100, 100
ZERO G6D, 95,  99,  99, 107, 103, 100

SPEED 5
```

Es marquen les posicions que es consideraran com a zero-point. S'assigna a la velocitat el valor 5. Es recomana no utilitza velocitats superiors a 10 per les diferents pràctiques a realitzar

```
'== motor power on  ============================
MOTOR G24

'================================================
'================================================

'Main procedure
```

```
INICI:
      'Inicialització

      GOSUB standard_pose
      ordre = 0
      quantitat_mov = 0
      GOTO REB_VALOR1
```

Es porta el robot a la posició estàndard. Seguidament s'inicialitzen les variables ordre i quantitat de moviment. Finalment es passa al següent subprograma.

```
REB_VALOR1:
      'Es rep el primer valor (ordre) i es comprova que sigui una ordre.
      En cas afirmatiu s'emmagatzema i s'envia OK, en cas contrari es
      torna a rebre.
      valor_reb = 0
      ERX 9600,valor_reb,REB_VALOR1
      IF valor_reb < 1 OR valor_reb > 3 THEN
            GOTO REB_VALOR1
      ENDIF

      ordre = valor_reb
      DELAY 1000
      ETX 9600,OK
      GOTO REB_VALOR2
```

Primer s'inicialitza la variable valor_rebut a 0, Després és captat el valor a través del port sèrie i s'emmagatzema a la variable valor_reb. En cas que no es rebi cap valor es tornarà a iniciar el procés de recepció. Una vegada captat un valor, es farà un petit filtratge descartant els valors que no siguin cap ordre (1,2,3). Si el valor no és correcte es tornarà a iniciar el subprograma. Si es correcte s'emmagatzemarà a la variable ordre i després d'una pausa s'enviarà el valor OK pel port sèrie. En darrer lloc, espassarà al següent subprograma.

```
REB_VALOR2:
      'Es rep el tercer valor (quantitat de vegades a realitzar el
      moviment) i es comprova que està en uns límits prefixats. En cas
      afirmatiu s'emmagatzema i s'envia OK, en cas contrari es torna
      a rebre.
      valor_reb = 25
      ERX 9600,valor_reb,REB_VALOR2
      IF valor_reb > LIMIT_MOV THEN
            GOTO REB_VALOR2
      ENDIF

      DELAY 1000
      quantitat_mov = valor_reb
      ETX 9600,OK
      GOTO ELECCIO_MOV
```

Primer s'inicialitza la variable valor_rebut a 0, Després és captat el valor a través del port sèrie i s'emmagatzema a la variable valor_reb. En cas que no es rebi cap valor es tornarà a iniciar el procés de recepció. Una vegada captat un valor, es farà un petit filtratge descartant els valors que superin el límit

preestablert de moviments plausibles(12). Si el valor no és correcte es tornarà a iniciar el subprograma. Si es correcte s'emmagatzemarà a la variable ordre i després d'una pausa s'enviarà el valor OK pel port sèrie. En darrer lloc, espassarà al següent subprograma.

```
ELECCIO_MOV:
	'Segons el valor de l'ordre es realitzarà algun dels moviments
	preestablerts i un cop realitzat s'enviarà el RESET per reiniciar el
	procés de Matlab.
	IF ordre = 1 THEN
		GOSUB mov_caminar
	ELSEIF ordre = 2 THEN
		GOSUB mov_girar_dreta
	ELSEIF ordre = 3 THEN
		GOSUB mov_girar_esq
	ENDIF

	DELAY 1500
	ETX 9600, RESET
	GOTO INICI
```

Segons el valor de la ordre capturada a través del valor sèrie. En cas e ser 1 es passarà al subprograma de moviments de caminar. Si és un 2 es passa al subprograma del moviment de gir a la dreta, mentre que si és un 3 es passarà al moviment de gir a l'esquerra.

Una vegada realitzats els moviments es farà una pausa de 1 segon i mig. Seguidament s'enviarà a través del port sèrie el valor RESET i posteriorment es reiniciarà el programa.

```
	'=================================================
	'=================================================

mov_caminar:
	'Realització del moviment de caminar

	passos = 1
	FOR passos = 1 TO quantitat_mov
		GOSUB caminar
		DELAY 700
	NEXT passos
	RETURN


mov_girar_dreta:
	'Realització d'un gir cap a la dreta

	girs = 1
	FOR girs = 1 TO quantitat_mov
		GOSUB turn_right
		GOSUB standard_pose
		DELAY 700
	NEXT girs
	RETURN


mov_girar_esq:
```

```
        'Realització d'un gir cap a la dreta

        girs = 1
        FOR girs = 1 TO quantitat_mov
              GOSUB turn_left
              GOSUB standard_pose
              DELAY 700
        NEXT girs
        RETURN
```

Una vegada escollit el tipus de moviment que s'haurà de realitzar simplement es faran tants moviments (passos/girs) com la variable quantitat_mov indiqui. Entre un pas/ gir i el següent es farà una breu pausa de 0.7 segons. Una vegada realitzats els moviments que pertoqui es tornarà al subprograma ELECCIÓ_MOV.

```
    '================================================
    '================================================
standard_pose:
        SPEED 5
        MOVE G6A,100,  76,  145,  93,  100,  100
        MOVE G6D,100,  76,  145,  93,  100,  100
        MOVE G6B,100,  30,  80,  100,  100,  100
        MOVE G6C,100,  30,  80,  100,  100,  100
        WAIT
        RETURN
```

Es defineixen els valors de la posició estàndard (posició inicial i de seguretat).

```
        '================================================
        '================================================
caminar:
        'Inclinar-se a la dreta
        SPEED 5
        MOVE24   85,   71,  152,   91,  112,   60,  100,   40,   80,     ,     ,     ,
        100,   40,   80,     ,     ,     ,  112,   76,  145,   93,   92,   60,

        ' 'Aixecar peu esquerre
        SPEED 10
        MOVE24   90,  107,  105,  105,  114,   60,   90,   40,   80,     ,     ,     ,
        100,   40,   80,     ,     ,     ,  114,   76,  145,   93,   90,   60,

        'Avançar peu esquerra + 'Baixar peu esquerre
        MOVE24   90,   56,  143,  122,  114,   60,   80,   40,   80,     ,     ,     ,
        105,   40,   80,     ,     ,     ,  113,   80,  145,   90,   90,   60,
        MOVE24   90,   46,  163,  112,  114,   60,   80,   40,   80,     ,     ,     ,
        105,   40,   80,     ,     ,     ,  112,   80,  145,   90,   90,   60,

        'Inclinar-se a l'esquerra + Aixeca peu dret
        SPEED 5
        MOVE24  100,   66,  141,  113,  100,  100,   90,   40,   80,     ,     ,     ,
        100,   40,   80,     ,     ,     ,  100,   83,  156,   80,  100,  100,
        MOVE24  113,   78,  142,  105,   90,   60,  100,   40,   80,     ,     ,     ,
        100,   40,   80,     ,     ,     ,   90,  102,  136,   85,  114,   60,

        'Avançar peu dret
```

```
      SPEED 7
      MOVE24 113,  76, 145,  93,  90,  60, 100,  40,  80,    ,    ,    ,
      90,  40,  80,    ,    ,    ,  90, 107, 105, 105, 114,  60,

      'Baixa peu esquerre fins alçada peu dret
      SPEED 5
      MOVE24 108,  84, 135,  94,  86,  , 100,  40,  80,  ,  ,  ,  90,  40,
      80,  ,  ,  ,  91,  88, 130,  98, 115,

      'standard_pose
      MOVE G6A,100, 76, 145, 93, 100, 100
      MOVE G6D,100, 76, 145, 93, 100, 100
      MOVE G6B,100, 30, 80, 100, 100, 100
      MOVE G6C,100, 30, 80, 100, 100, 100
      WAIT

      RETURN
```

Per a realitzar el moviment de caminar del Visual Servoing es fa servir una edició del realitzat en les pràctiques. D'aquesta manera s'aconsegueix que el desplaçament endavant sigui el mínim possible i es tingui més precisió en el control.

```
      '================================================
      '================================================
turn_right:

      HIGHSPEED SETON

      MOVE G6D, 100,  85, 164,  99, 100
      MOVE G6A, 100,  67, 126,  87, 105
      WAIT

      MOVE G6D, 100,  76, 145,  93, 100
      MOVE G6A, 100,  76, 145,  93, 100
      WAIT

      HIGHSPEED SETOFF

      RETURN

      '================================================
      '================================================
turn_left:

      HIGHSPEED SETON

      MOVE G6A, 100,  84, 161,  98, 100
      MOVE G6D, 100,  68, 129,  88, 105
      WAIT

      MOVE G6A, 100,  76, 145,  93, 100
      MOVE G6D, 100,  76, 145,  93, 100
      WAIT

      HIGHSPEED SETOFF
```

RETURN

Per a realitzar els moviments de gir tant a dreta com a esquerra es fan servir unes comandes diferents a les usades en les pràctiques anteriors també buscant més precisió. En aquest cas es fas ús de la velocitat màxima per portar el robot a un punt inestable i retornar immediatament. Degut a la fricció amb el terra per a realitzar aquest moviment, el robot fa gira. Tal i com es mostra, en aquest cas els moviment només impliquen les cames ja que els grups B i C dels braços no apareixen.